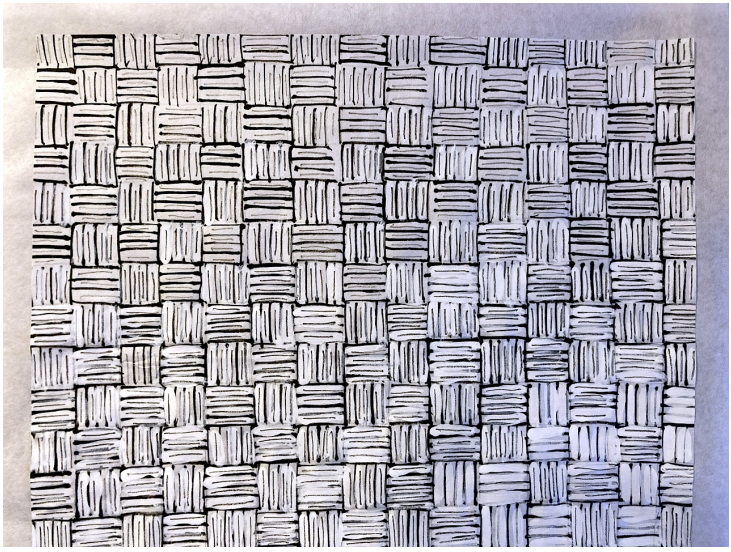
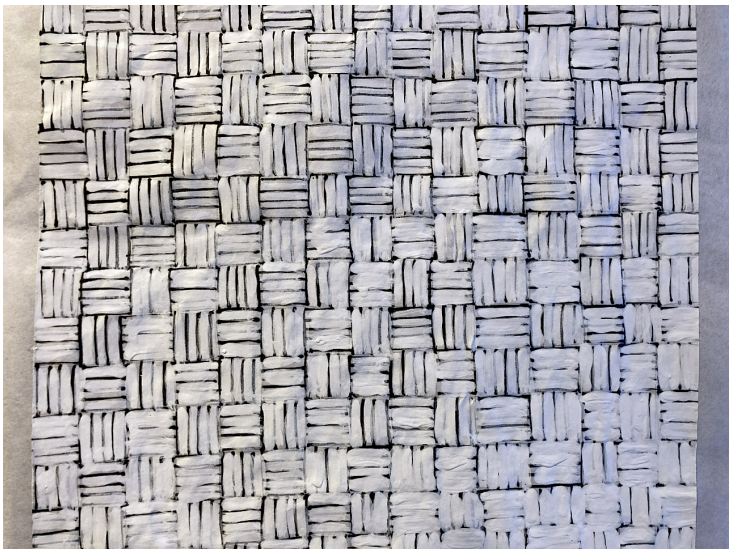
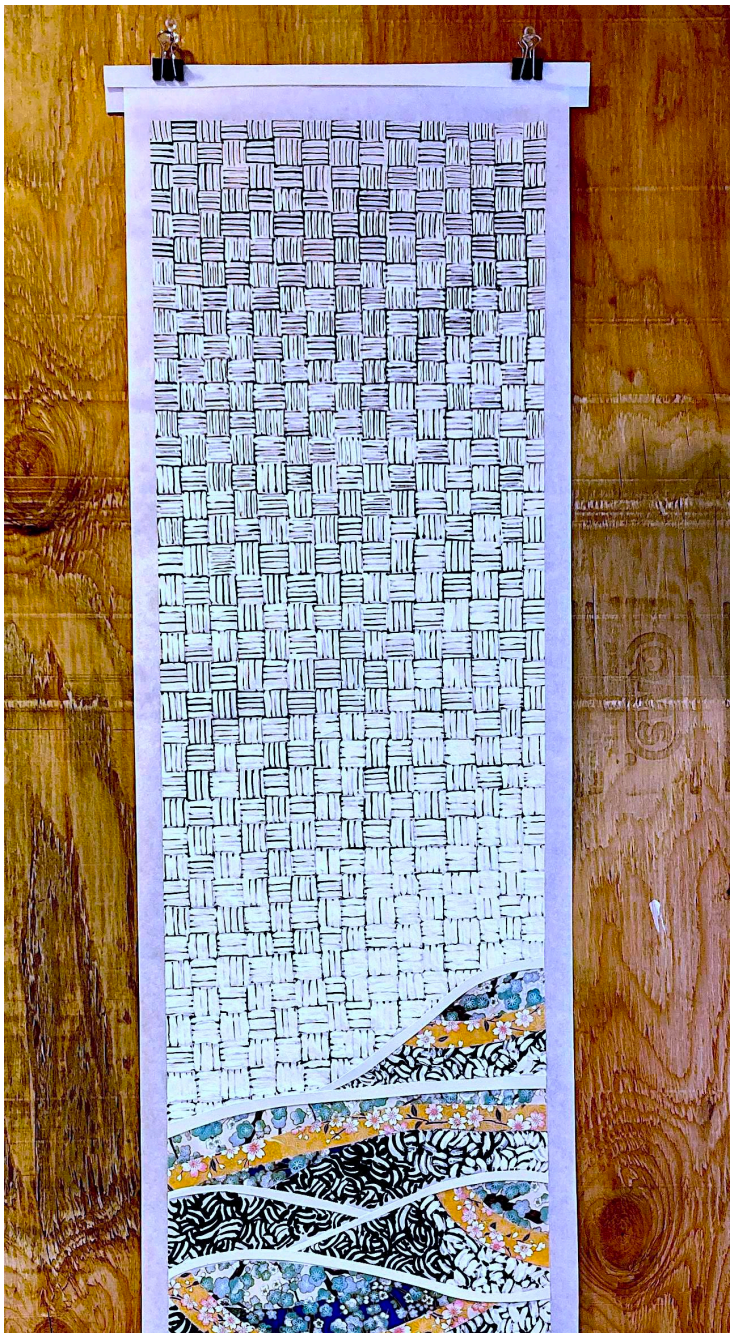
intellectual athleisurewear

Russian River (2022)

Russian River (2022)







11 inches by 43 inches. Ink and gesso and pencil and chiyogami on paper.

2022-03-19

things I have written about elsewhere #20220330

Serving map tiles to yourself using Protomaps...

Serving map tiles to yourself using Protomaps and iOS

This was **originally published on the SFO Museum Mills Field weblog** `https://millsfield.sfomuseum.org/blog/2022/03/30/swifter-protomaps/` , in March, 2022.

Airline bag: Cathay Pacific Airways. Vinyl, plastic, metal, paint. Gift of the William Hough Collection, SFO Museum Collection. ***2004.061.008***

swifter-protomaps `https://github.com/sfomuseum/swifter-protomaps` is a **Swift** `https://swiftlang.org` -language package that exports a single function which implements code to serve **Protomaps** `https://protomaps.com` tile databases from a **httpswifter/swifter** `https://github.com/httpswift/swifter` web server. For many people, that sentence might sound like **word salad** `https://en.wikipedia.org/wiki/Word_salad` so let's go through each one of those items individually.

1. ***Swift*** `https://swiftlang.org` *is a programming language used to write applications for the MacOS and iOS operating systems.*

2. ***swifter*** `https://github.com/httpswift/swifter` *is a library that allows developers to write custom web servers, as standalone applications or as a part of a larger program, written in Swift.*

3. ***swifter-protomaps*** `https://github.com/sfomuseum/swifter-protomaps` *is a library that defines functionality for serving Protomaps database files from a* `swifter` *web server.*

4. ***Protomaps*** `https://protomaps.com/` *is a framework for bundling and dynamically styling map data using static database files, sourced from the* ***OpenStreetMaps*** `https://openstreetmaps.org` *project.*

`swifter-protomaps` bundles the **100+ lines of code** `https://github.com/sfomuseum/swifter-protomaps/blob/main/Sources/SwifterProtomaps/SwifterProtomaps.swift` necessary to serve Protomaps database files in a `swifter` web server in to 4-6 lines of code. In short, it is what's often referred to as a "helper" function. It is not fancy, or sophisticated, but we think it is useful.

```
import Swifter
import SwifterProtomaps

let root = URL(string: "/path/to/pmtiles")
let port: in_port_t  = 9000
```

```
let opts = ServeProtomapsOptions(root: root)
opts.AllowOrigins = "*"
opts.AllowHeaders = "*"

let server = HttpServer()

server["/pmtiles/:path"] = ServeProtomapsTiles(opts)
server.start(port)

// Error handling has been removed from the example above for the sake of brevity.
```

We have published this code as part of our ongoing commitment to releasing and promoting "**small focused tools** https://millsfield.sfomuseum.org/blog/2020/08/04/small-tools/ ":

> The cultural heritage sector needs as many small, focused tools as it can produce. It needs them in the long-term to finally reach the goal of a common infrastructure that can be employed sector-wide. It needs them in the short-term to develop the skill and the practice required to make those tools successful. We need to learn how to scope the purpose of and our expectations of any single tool so that we can be generous of, and learn from, the inevitable missteps and false starts that will occur along the way.

But why was any of this necessary in the first place?

Airline bag: Canadian Pacific Airlines. Synthetic textile, vinyl, metal, paint. Gift of Thomas G. Dragges, SFO Museum Collection. ***2001.146.093***

Last year, in the **Interactive maps of SFO from 1930 to 2021 (and beyond) at the T2 SkyTerrace** `https://millsfield.sfomuseum.org/blog/2021/11/19/skyterrace/` blog post I wrote:

> The first versions of the [SkyTerrace interactive map] application also used a very detailed, but ultimately distracting, cartography to depict the present day airport and surrounding Bay Area. Once the decision was made to refactor the touch-based application in to something that could be controlled remotely I also took the opportunity to revisit the cartographic elements that frame the historical maps. To start, the detailed road networks of the San Francisco Bay Area were de-emphasized in favour of the Bay itself.

The coastline for the San Francisco Bay Area is derived using data from the **OpenStreetMap** `https://wiki.openstreetmap.org/wiki/Coastline` project. That data is encoded as a **GeoJSON** `https://tools.ietf.org/html/rfc7946` record (700kb) which is bundled with the SkyTerrace interactive map application and rendered using the **Leaflet** `https://leafletjs.com/` mapping framework. This is a pretty common approach for web-based mapping applications and is orders (and orders) of magnitude simpler than rendering and storing individual map tiles for an area the size of the Bay Area from zoom levels 12 to 20.

Airline bag: United Air Lines. Synthetic textile, plastic, metal, string, ink. Gift of Thomas G. Dragges, SFO Museum Collection. ***2009.028.037 a b***

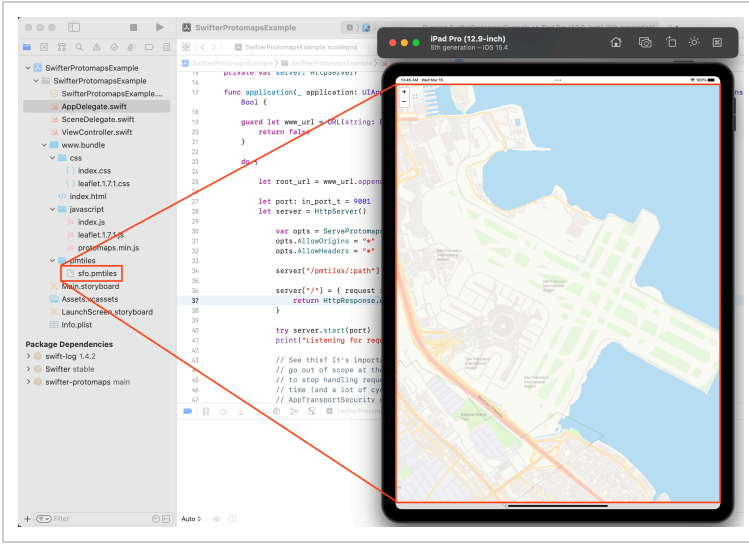
This approach worked great until iOS 15, which introduced changes to the way that vector data (for example GeoJSON data) is rendered by the underlying operating system and suddenly a lot of things started breaking, including the SkyTerrace interactive map application. We figured this out through a process of trial and error and by trying to find reports of other people having similar problems, eventually landing on a bug report filed with Apple titled **Safari crashes when GPU Process: Canvas Rendering is enabled with large paths** `https://bugs.webkit.org/show_bug.cgi?id=231157` .

The San Francisco Bay Area coastline is a "large path" and we were able to prevent crashing bugs by not including it in our application's interface. With that knowledge I implemented some temporary

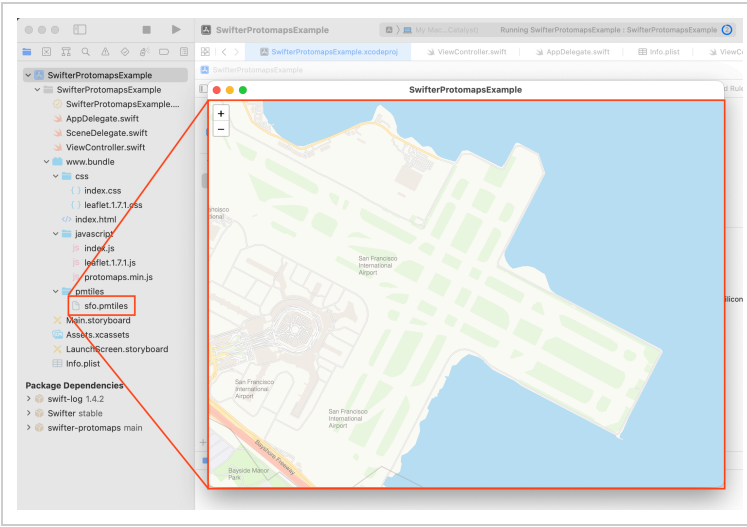
workarounds and waited patiently for the bug to be fixed and rolled out to Safari (or more accurately WebKit which is the Safari web browser's rendering engine). That happened **earlier this year** `https://trac.webkit.org/changeset/284079/webkit` but those changes have not fixed our specific problem. Frustratingly, the bug has been fixed in both mobile and desktop versions of Safari but persists in instances of **WebKitView** `https://developer.apple.com/documentation/webkit/wkwebview` which is the iOS component used to embed and render web content in iOS applications. For a longer discussion of how we're using WebKitView to develop applications have a look at the **iOS Multi-screen Starter Kit** `https://millsfield.sfomuseum.org/blog/2020/11/18/multiscreen/` blog post.

The "large path" of the San Francisco Bay Area coastline, rendered using Protomaps.

Following a few **false starts** `https://www.newyorker.com/magazine/1994/07/11/forty-one-false-starts` I eventually realized that if the SkyTerrace interactive map application could be updated to run its own web server as a background task then that web server could deliver tiles (to the web application running in a WebKitView component) for the San Francisco Bay Area coastline using a Protomaps database file and render them using the **protomaps.js** `https://github.com/protomaps/protomaps.js` library. The Protomaps database file for the San Francisco Bay Area coastline is approximately 135MB which is over one hundred times larger than the GeoJSON file we've been using to draw the coastline but it's also a fraction of the size required to store individual map tiles for the same area and only takes a couple of minutes to produce using the Protomaps **large area map tool** `https://protomaps.com/downloads/large_map` .



To help demonstrate why we think this is so exciting, we have released **swift-protomaps-example** `https://github.com/sfomuseum/swifter-protomaps-example` which is a simple iOS (and Mac Catalyst) application to demonstrate use of the `sfomuseum/swifter-protomaps` package. The application displays an interactive map of the San Francisco International Airport (SFO) and surrounding area using data from the **OpenStreetMaps** `https://openstreetmaps.org` project. That data is stored in a Protomaps database file (2MB) which is bundled with the application which means it also works offline.



Aside from solving an immediate technical problem, we are excited about how this approach might be applied to future projects and we hope you will be too.

2022-03-30

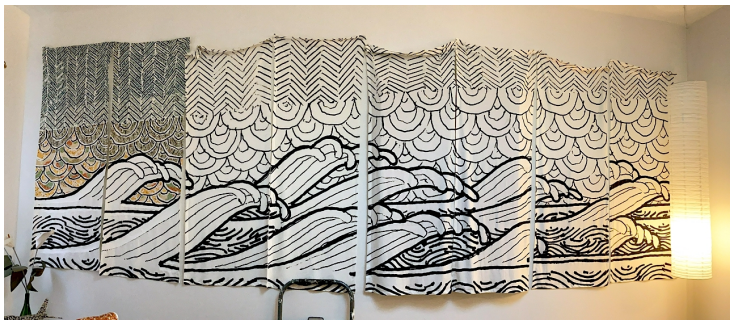
front running nostalgia

Ocean Beach in Eight Panels (2021-2022)

Ocean Beach in Eight Panels (2021-2022)

Digital print, ink, gesso, chiyogami and katazome on fabric. 186 inches by 63 inches (approximately). Large version **here** `/weblog/2022/04/27/frontrunning/images/ob-8-008-2560.jpg` .

The original drawing that this was derived from is about 11 inches by 3 inches. **The same but different.** https://aaronland.info/weblog/2020/10/27/uniformity/#same



By now some people might have noticed that the offsets for the last panel are incorrect. Don't worry if you didn't notice because I didn't either until just before I took this picture.

That was discouraging. It was also good practice for making the same mistake twice in a row.

It's a bit hard to see here but after all that I still managed to get the measurements on a properly-offset last panel wrong.

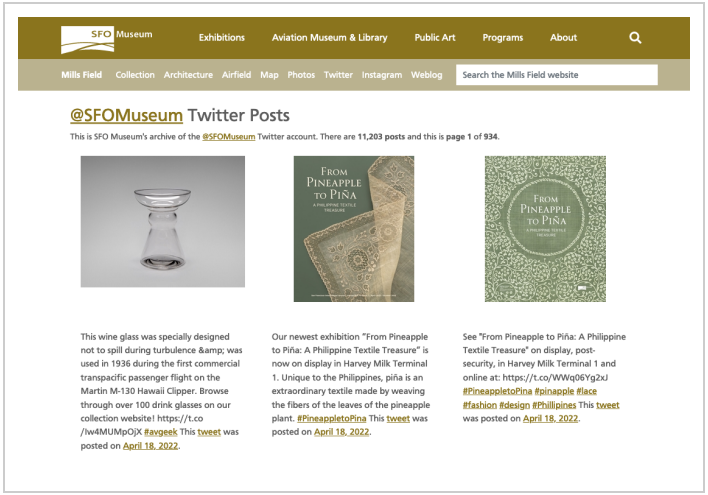
Eventually it worked out. If the **map=yes** `http://mapequalsyes.stamen.com/` project was a love song to Dave McKean's **Cages** `https://en.wikipedia.org/wiki/Cages_(comics)` this one owes a debt to the work of **Lynn Varley** `https://en.wikipedia.org/wiki/Lynn_Varley` which is well known within her professional circles but deserves to be more widely celebrated.

2022-04-27

things I have written about elsewhere #20220504

Archiving social media accounts at SFO Museum –...

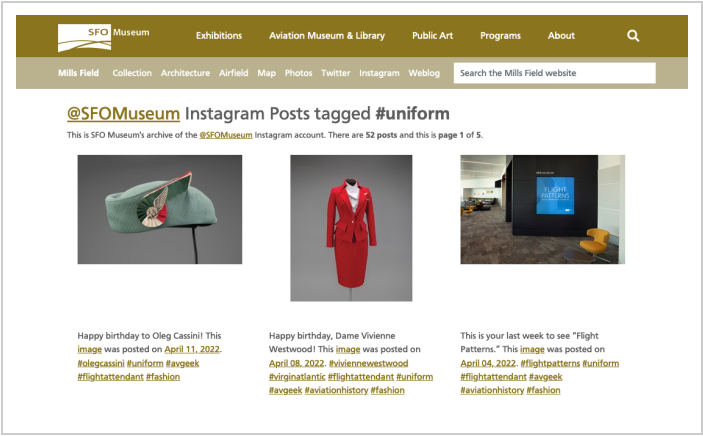
Archiving social media accounts at SFO Museum – Take three



https://millsfield.sfomuseum.org/twitter/

This was **originally published on the SFO Museum Mills Field weblog** `https://millsfield.sfomuseum.org/blog/2022/05/04/sociamedia/` , in May, 2022.

There wasn't meant to be **a two-year hiatus between updates** `https://millsfield.sfomuseum.org/blog/2020/10/28/socialmedia/` to the local archive of SFO Museum social media accounts but that's what happened. In mid-April we finally downloaded fresh exports from Twitter and Instagram and updated the tools we use to process those data and now there are over **a decade's worth of posts** `https://millsfield.sfomuseum.org/socialmedia/` from each service on the Mills Field website. The process is still not fully automated but we are a few steps closer to acheiving that goal.

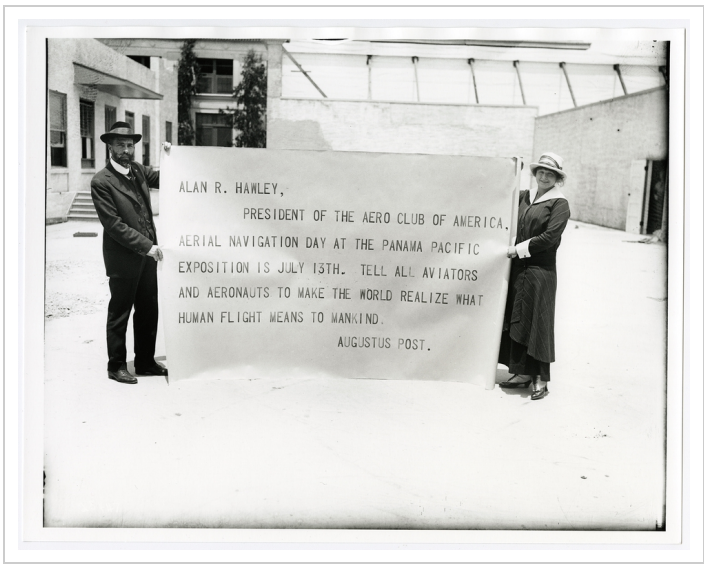


https://millsfield.sfomuseum.org/instagram/

Depending on how you feel about the impending sale of Twitter these kinds of processes can take on an added urgency. Fundamentally, though, we believe that this - creating our own local archive of our social media accounts - is an important practice regardless of the internal drama any given service is experiencing. These third-party services that we use offer many benefits but too often we forget that they are not necessarily built for for longevity. Importantly it's not necessarily their responsibility either. In the **first blog post about archiving SFO Museum's social media accounts** `https://millsfield.sfomuseum.org/blog/2019/03/06/twitter/` I wrote:

> SFO Museum makes a point of being active on a number of social media platforms. We have accounts on Twitter and Facebook and Instagram and a full-time staff member – Bao Li – whose job it is to craft and curate posts related to the museum and its collection and share them far and wide on the internet. It is important to recognize that Bao's work is not simply "non-institutional contextualization of digitized collection objects" but an important contribution, one that is central to the museum's mission ... but we haven't done a great job of "capturing" or "archiving" any of it.

So long as there is a way for SFO Museum to export the things that it posts on a service we can and should take on some of the burden of preserving those efforts for posterity. That is, after all, the business of museums and libraries and archives.



Photograph: Panama-Pacific International Exposition. Gelatin silver print. Gift of Edwin I. Power, Jr. and Linda L. Liscom, printing funded by San Francisco Aeronautical Society, SFO Museum Collection. ***2015.040.060 a***

As part of that effort we have updated the Go package we use to process Instagram archives to include a new tool called **derive-media-json** `https://github.com/sfomuseum/go-sfomuseum-instagram#derive-media-json` . This is a command line tool to derive an abbreviated "media.json" file from a "contents/posts-(N).html" file as published by the Instagram export tool, circa April, 2022.

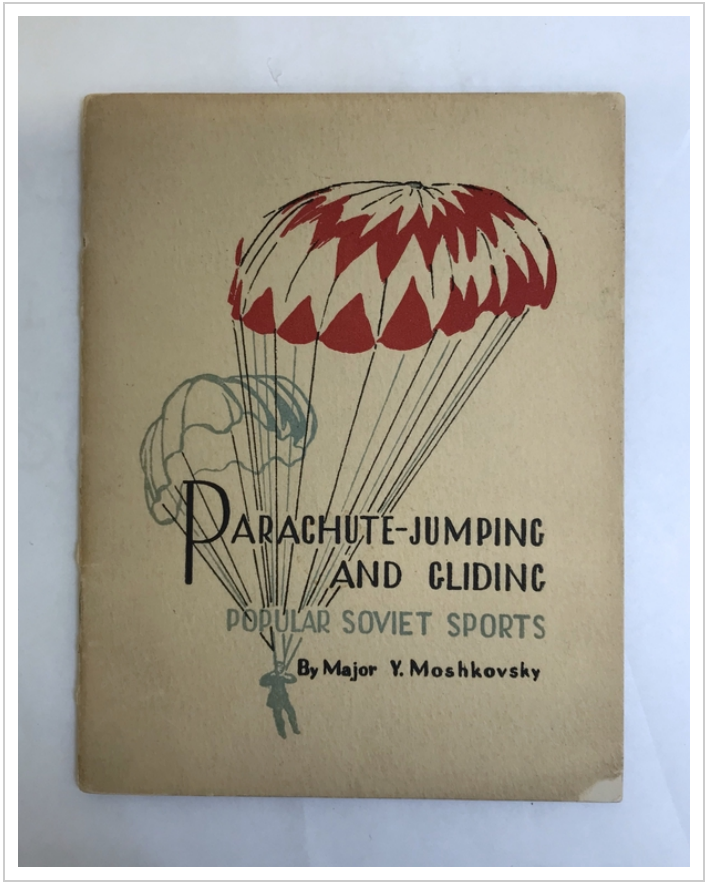
Previous Instagram export data bundles (circa October, 2020) used to provide one or more "media-(N).json" files that contained machine-readable properties for working with Instagram exports. This tool attempts to reconstruct that data derived from HTML markup and outputs the results as JSON to STDOUT.

For example:

```
$> bin/derive-media-json /usr/local/instagram-export/contents/posts_1.html

[
  {
   "path": "media/posts/201502/1209467_621332467997055_325446168_n_17841739630062499.jpg"
   "taken_at": "2015-02-26T15:07:00Z",
   "caption": "\"Making art is like escaping to find peace of mind.\" -Lee Kang Hyo (b. 1
   }
  },
  {
   "path": "media/posts/201502/10986292_690732684371255_1179212910_n_17841739627062499.jp
   "taken_at": "2015-02-13T15:15:00Z",
   "caption": "Gorgeous and golden details emphasize the exotic elements on this 1860s-70
  }
   ... and so on
```

It is expected that this tool is brittle precisely because it is parsing non-structured data observed at a moment in time. This tool has been demonstrated to work with Instagram exports as published in April, 2022 but there are no guarantees that this tool will work with future (or past) Instagram exports. This tool should not need to exist but until equivalent machine-readable data is published by Instagram it will have to do.



Parachute-jumping and gliding: popular soviet sports. Paper, ink. The Tony Bill Aviation Library Collection, SFO Museum Collection. ***2000.160.1301***

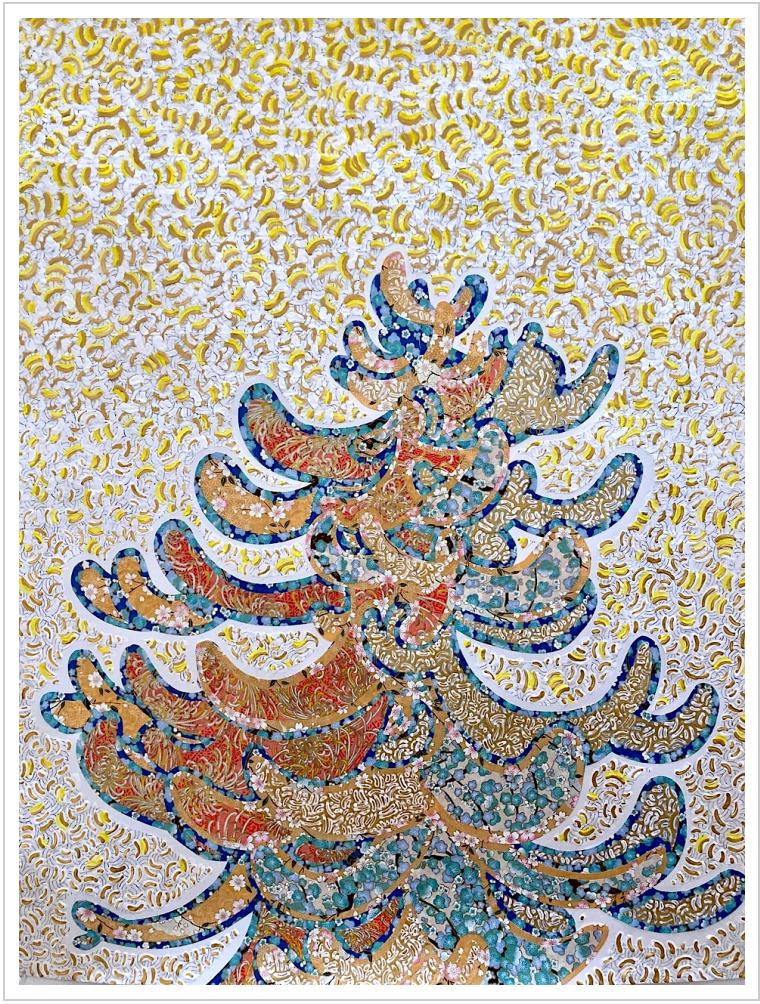
What you do with this machine-readable data is up to you. The `derive-media-json` tool is part of our ongoing effort to create **small focused tools** `https://millsfield.sfomuseum.org/blog/2020/08/04/small-tools/` that do one thing well and to share those tools with others to save them time while fostering a **culture of generosity** `https://www.newsweek.com/new-wisdom-web-107953` .

2022-05-04

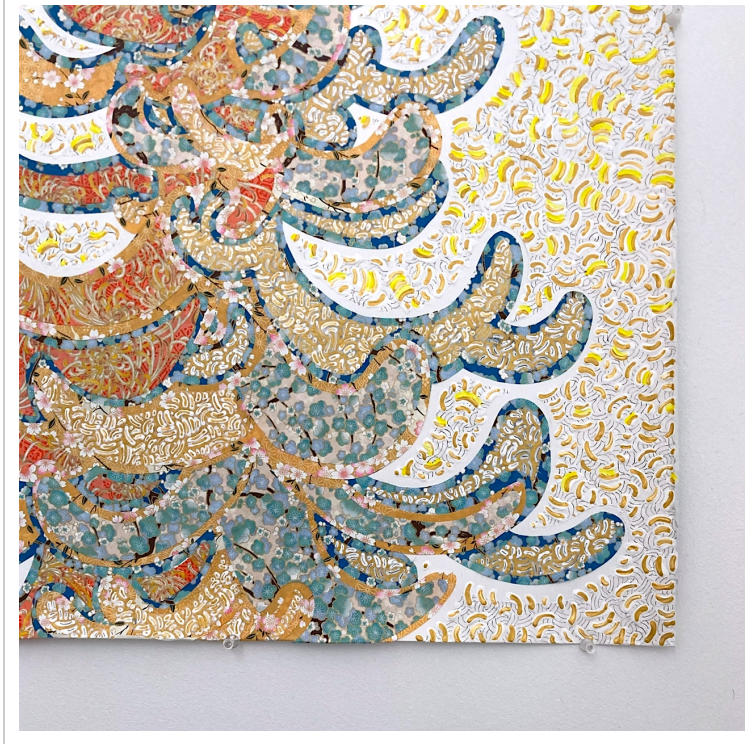
trying to remember

Redwood #5 (2022)

Redwood #5 (2022)



Acrylic and graphite and chiyogami on paper. 24" x 30".



2022-05-06

style transfers for metaphors

Bollard (2022)

Bollard (2022)

Chiyogami and gesso and googley eyes on insulation piping. 8" x 8" x 8" (approximate).

2022-05-31

ferris wheel halos

Bollard (2022) #2

Bollard (2022) #2

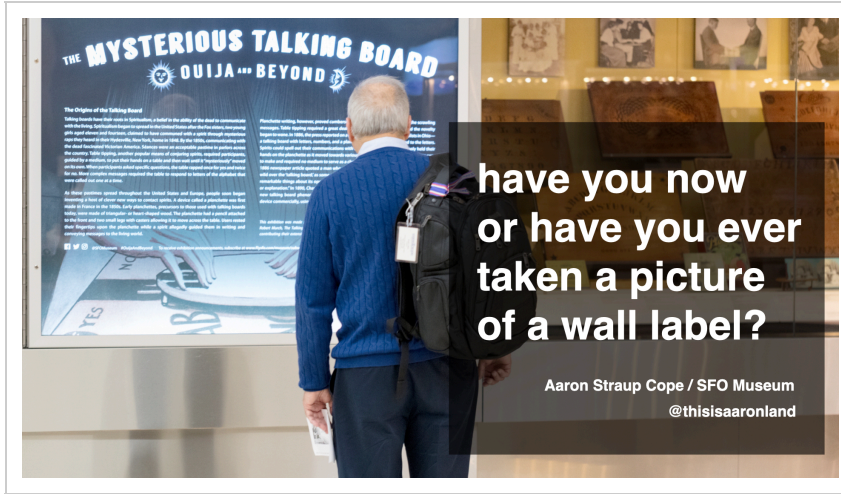
Katazome and acrylic and googley eyes on insulation piping. 8" x 8" x 8" (approximate).

2022-06-13

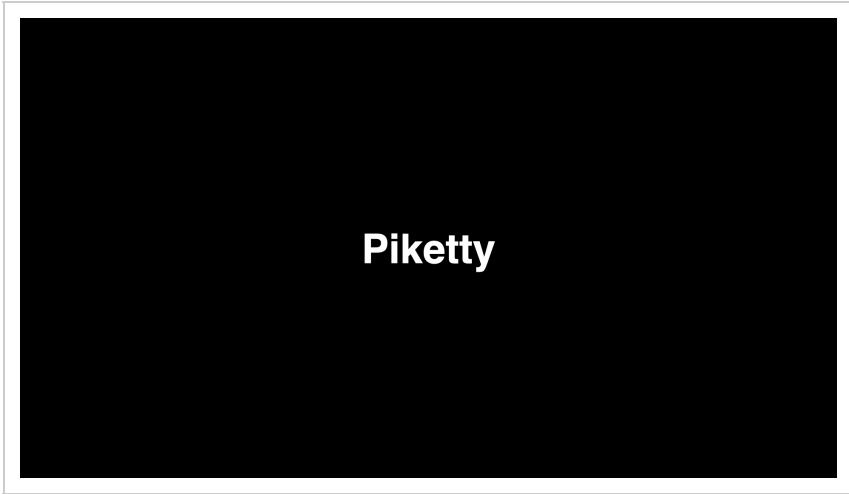
sometimes expectations happen to you

have you now or have you ever

have you now or have you ever



In April, I was asked to speak a class of museum studies students at the ***University of San Francisco*** `https://www.usfca.edu/arts-sciences/graduate-programs/museum-studies` *. Apparently there is still interest, after all these years, in hearing about The Pen. As it happens I had recently done a internal presentation at work about The Pen. I used that presentation as the basis for the museum studies class and then extended it further to make the link between some more* ***recent work at SFO Museum*** `https://millsfield.sfomuseum.org/blog` *to address many of the same basic (ongoing) problems.*



I'd like to start with the French economist **Thomas Piketty** https://en.wikipedia.org/wiki/Thomas_Piketty . In 2013 he published a book titled **Capital in the Twenty-First Century** https://en.wikipedia.org/wiki/Capital_in_the_Twenty-First_Century . It is a comprehensive study of wealth distribution and inequality since the 1800s. One of the things he notes in the book is that his analysis would not have been possible had the French authorities, following the revolution in 1789, not started keeping comprehensive tax records year over year. Those records had to wait over 200 years before they would be use to support Piketty's argument. Those records did not "go viral" or generate any immediate "reward" after the years in which they were collected.

I wanted to start with Piketty's example because it demonstrates the value of persistance; the value of persisting something beyond the limits of the present's imagination. I wanted to start with this idea of persisting things because one of the genuinely novel things about the moment we are living through is that we have the means to store and to replicate, to persist, data at a scale that has literally never been possible before. Only a very few organizations, like the Catholic Church for example, have had the means to proative actively keep "all the things" for thousands of years. It is now possible for basically everyone else to do the same, or something like it, digitally.

Digital archiving is not a practice without risks or unintended consequences so it remains to be seen whether it has a thousand-year lifespan. My point is not that digital storage is the solution to all our problems. My point is only that where before there was only the gaping maw of nothing, of the past being obliterated by the negligence of the present or the malice of the victors or both, now there is something.

In more recent year, that persistance has been coupled with better and better tools for retrieving that data. "The past keeps changing", wrote the historian **Margaret MacMillan** `https://en.wikipedia.org/wiki/Margaret_MacMillan` , "because we keep asking questions of it." The advances in storage and retrieval technologies that we've seen just in our lifetimes are astonishing when compared to how we were able to ask questions of the past before their introduction. Importantly these tools have not only changed how we are able to ask question but who is able to ask them; namely more than the few.

And these changes are what give rise to higher order processes; the applications and functionalities that become possible when fast and cheap storage and retrieval of "stuff" can be taken for granted and dissolve in to the background.

As those higher order processes are normalized they start to be taken for granted and people's baseline expectations begin to change. For example, consider the way people now just assume it's possible to carry a tiny computer in their pocket and use it to connect to other computers thousands of miles away to answer a question or order cat litter. I've just described Wikipedia and Amazon, both of which are underpinned by an enourmous amount of resources devoted to storing and retrieving data.

Not infrequently these new found expectations start to impact everything else around them. People start to ask questions like "*If I can (x with a) then why can't I (y with b)?*" These questions sometimes betray a lack of understanding in what it takes to do "x". For example, the ability to ensure that cat litter is only ever a day away from you requires a mind-bogglingly sophisticated global logitics and warehousing infrastructure. But in world that celebrates the commodification of the possible these misunderstandings are not entirely unfair.

One consequence of these misunderstandings is that other people's changing expectations happen to you whether you are ready for them or not. Which raises an interesting question: Are expectations just the cost of doing business?

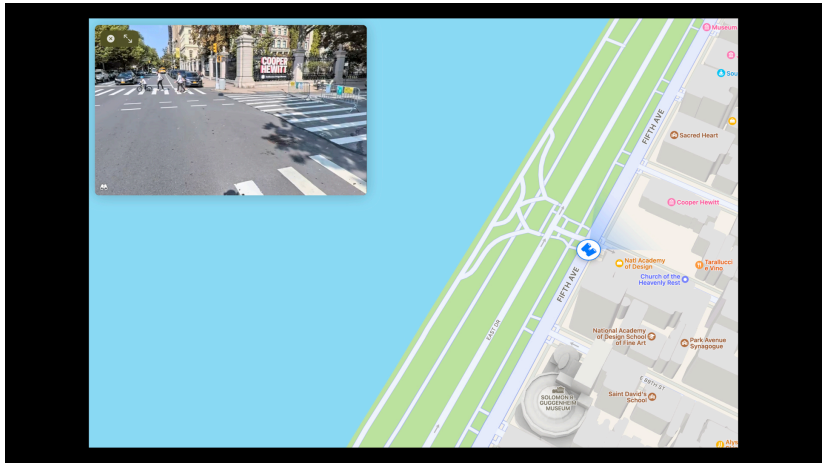
What I mean by that is does your enterprise aim to meet expectations simply as a way of garnening attention? Are people's changing expectations simply the zeitgeist that you cater to as a way to stay front of mind? The point of asking the question is not to say one approach is inherently better, or worse, than the other.

The point of asking the question is to better expose one's motivations. The reason for doing that, when we talk about museums, is because you just as easily replace the phrase "expectations" with "digital technologies" and ask the same questions. Is the role of digital technnologies in a museum to serve principally as an attractor, the thing that gets people through the door of an institution, or something else?

Again, there are a range of equally valid answers to that question. The problem I've observed is not that any two museums have different answers but that they are, typically, unable to articulate them in concrete terms. What happens, instead, is that everyone

brushes their respective nuances, biases and purposes under a collective rug that we've labeled "digital". The consequence of which is that we all use the same term to speak about radically different things and then wonder why we always seem to be talking at cross purposes.

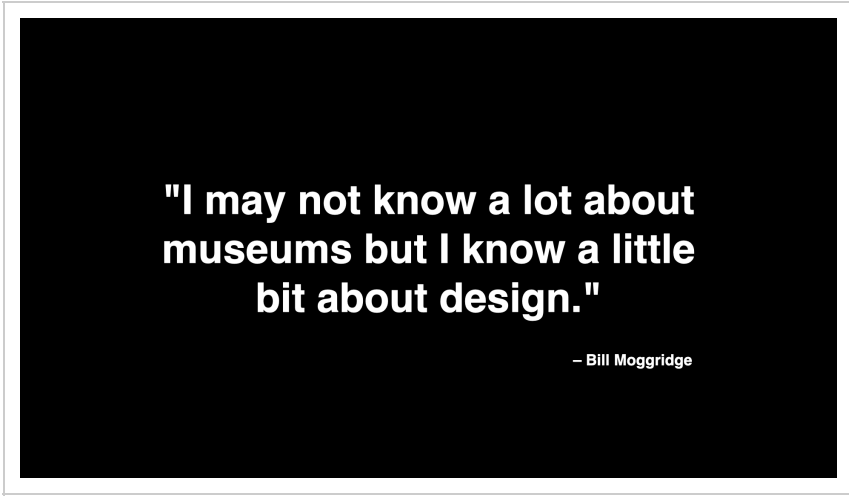
I want to suggest that perhaps the most important thing you can do now, as people thinking about using the words "museum", "professional" and "digital" in the same phrase, is to develop a personal understanding of what the point of incorporating digital technologies in to a museum is and then to find the language to express to others what those ideas are and are not. Do this as a favour both to yourselves and to your peers.



The **Cooper Hewitt** `https://cooperhewitt.org` , the Smithsonian's National Design Museum in New York City, is a good study for questions about attractors. The Cooper Hewitt is literally a block away from the **Guggenheim** `https://www.guggenheim.org` but, for many people, it might as well be across the ocean. Here's a visual aid to make the point. The Guggenheim, with its easily recognizable cylindrical building, can be seen in the bottom right of this screenshot. The little binocular icon, floating above the Guggenheim, is pointing at the Cooper Hewitt.

The problem facing the Cooper Hewitt, when it closed its doors for renovations in 2011, was that most people would never walk that extra block to visit the museum. Many didn't even know it was there in the first place. Even if they did what they encountered was a very old and foreboding, mansion. You'd be forgiven for thinking it was the private residence because it was; this was Andrew Carnegie's home. So the problem facing the Cooper Hewitt was that it needed

something, an attractor, to help people overcome those barrier of simply walking through the door.



In 2010, the Smithsonian hired **Bill Moggridge** https://en.wikipedia.org/wiki/Bill_Moggridge to be the new director of the Cooper Hewitt. Bill was not a museum professional but rather a life-long designer having, among his many other accomplishments, designed the first laptop computer and helped found the design studio IDEO.

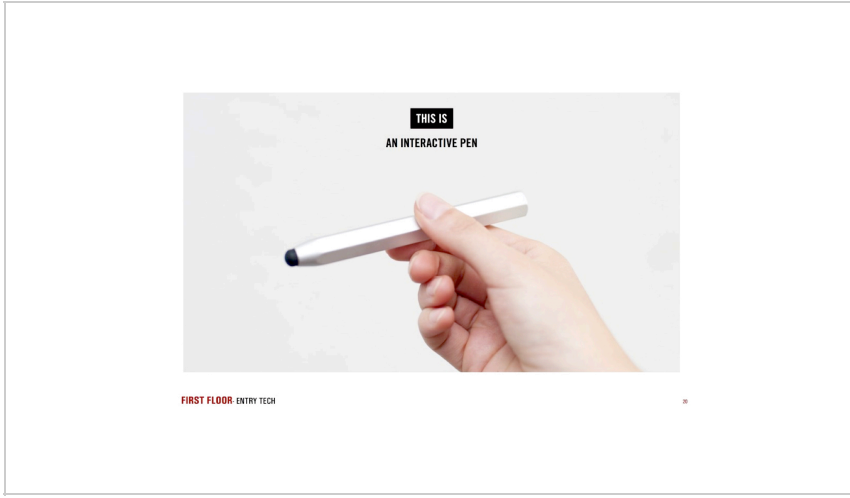
Bill was hired to address the problems I've just described. Bill was hired to help bring the museum in to the "digital age". Whether that phrase enjoyed the clarity of intention I've been arguing for remains unknown but it led Bill to hire **Seb Chan** https://www.sebchan.com/ to run the newly created **Digital and Emerging Media department** https://labs.cooperhewitt.org/ . In 2012 I would join that team.

The great luxury that design museums have is that everything, or at least everything made by humans, is designed. Whether people know it or not their use and experience of those "things" is directly related to, and reflected in, their design. The challenge for a design museum is to find a way to convey those relationships to people.

In 2011 Fast Company **published a profile piece about Bill** `https://www.fastcompany.com/1777623/mister-moggridge-has-mad-ambition` in which he is quoted as saying:

> “The movement in museums that I’ve observed,” he says, “is that it’s an experience where people learn by doing and not just looking. You can’t really do much with treasures that can only be looked at and have to be protected.”

This conversation about "doing and not just looking" and about how to manifest that idea in the museum was already in full swing when I arrived.



This is the slide that birthed The Pen.

The pitch was that every every visitor would be given digital stylus upon entry that would both double as their ticket, or proof of purchase, and act as a device to invite visitors to participate in the act of designing. This would be done by prompts to sketch and create objects on interactive tables that would be connected, digitally, to the entirety of the museum's collection. These digital interfaces would connect the objects on display in the galleries to all the other related objects that remained in storage. Importantly the stylus was meant to give visitors license to *touch* things in the museum. Not the objects themselves but something to break the traditional conveyor belt of passive knowledge consumption that defines so many other museum experiences.

The Pen, it was suggested, could also be used to "collect" objects by tapping a wall label and recording that object's unique identifier, using RFID technologies. Visitors could, in effect, bookmark objects

they found interesting during a visit and the museum would provides the means by which those bookmarks could be retrieved at a later date. The point was not so much to judge which object's people collected or to dictate what they might do with them so much as to simply provide the means by which they could.

The Pen was meant to offer two sets of interactions: First, it was to be an invitation for visitors to be active participants during their time in the museum by giving them the ability to create their own drawings and sketches in relation to the collection and license to touch the architecture of the museum itself. Second, it not only afforded people the means by which they might catalog, or remember, their visit but encouraged them to do so.

In many ways the first set of interactions, combined with the novelty of the Pen itself, served as the attractor I talked about earlier. The second set of interactions suggested a means by which we might approach some of the broader problems around museum visitation and the relationships people have with our collections.

If the larger goal is to promote and foster a practice of revisiting, which I will argue is the bedrock of the humanties and a celebration of cultural heritage, then digital technologies offer a unique opportunity: To simply be a way in which we might help people to remember their visit to the museum and to use those same tools as a place where the stories and anecdotes that define the objects we collect might also be shared.

We used to ask people whether they have had ever taken pictures of wall labels during a visit to a museum. Most people, even people who were not museum professionals, said yes. When asked whether they ever did anything with those pictures they all said "No".

What if, we asked, you never had to worry about taking a picture of a wall label again? What if, instead, the museum itself stored a

record of your interest in an object and provided the means to retrieve it at some later date? What are the "higher order processes", akin to the Wikipedias and Amazons I talked about earlier, that might be possible once remembering a visit to the museum could be taken for granted?

This is a video we made to demonstrate these ideas internally. It was filmed using a commercially available device, developed for inventory tracking. In the beginning we imagined it might be possible to take an existing off-the-shelf solution for recording objects using NFC tags and retrofit it to our needs.

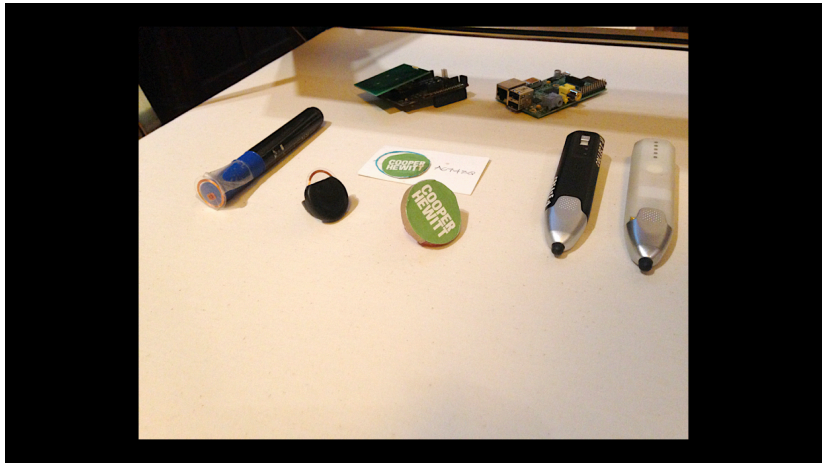
This was not the case as it turned out.

It took a lot of work, a lot of help from friends of the Museum and more-than-lunch-money from Bloomberg Philanthropies to turn that initial concept slide in to the finished product you see here.

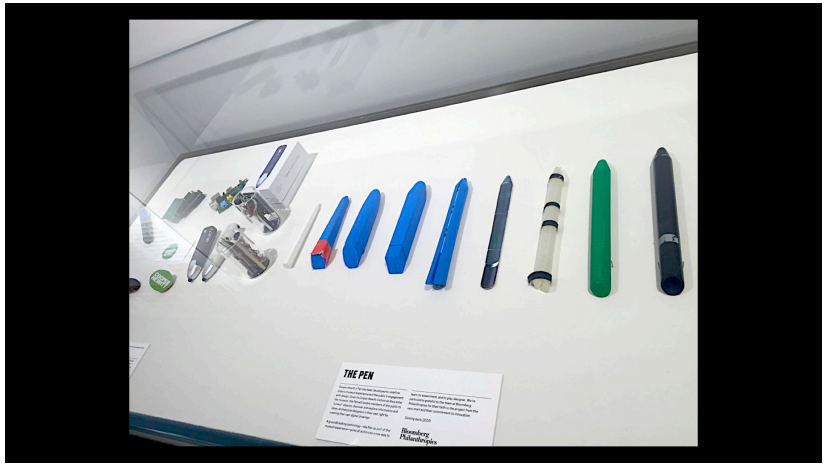
I suppose that's the bad news: It was really, really hard to make the Pen come to fruition. Hard in a way that some people still don't think was worth the effort. The good news is: It is possible to design and manufacture bespoke and sophisticated electronic hardware purpose-fit for a museum environment. I still find that exciting because it makes real the question: If this is possible what else could we do?

I also thought it was an important process for a design museum to experiece. To live the challenges and setbacks of taking the sketch of an idea and seeing it through to completion, through all the different economic and material considerations and the many actors that are necessary to mass-produce consumer products. That practice is the "business" of a design museum and what better way to help understand and convey its many nuances and complexities than by

doing it ourselves?

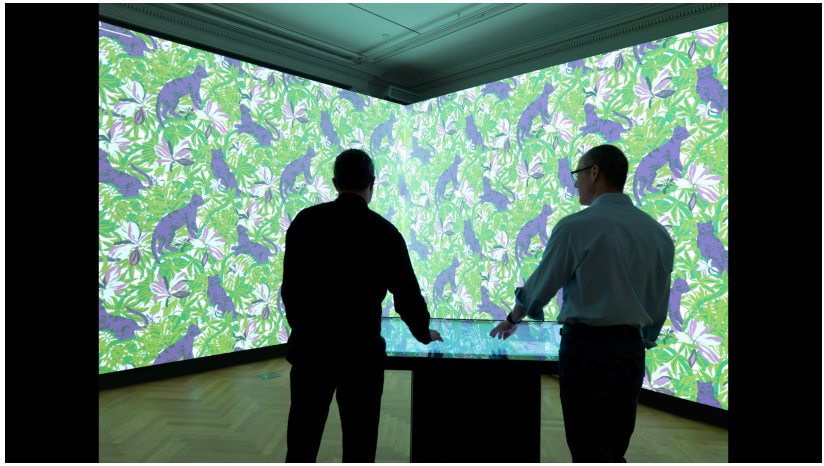


These are some of the prototypes we worked on in between the initial concept and the finished product. You can see the device we used for the initial concept video on the right. The company that made those devices was contracted to design and manufacture the electronics for the Pen.



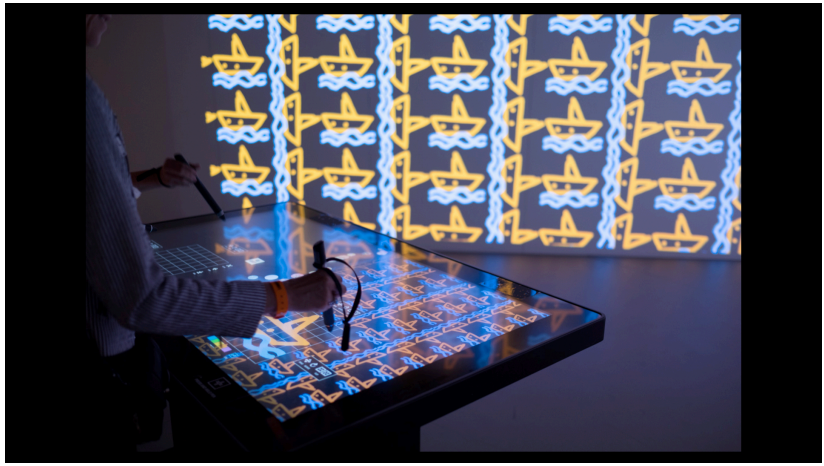
These are some other prototypes that show the iterations we went through as we approached the finished design.

Here's what the Pen and the interactive tables looked like in the galleries.



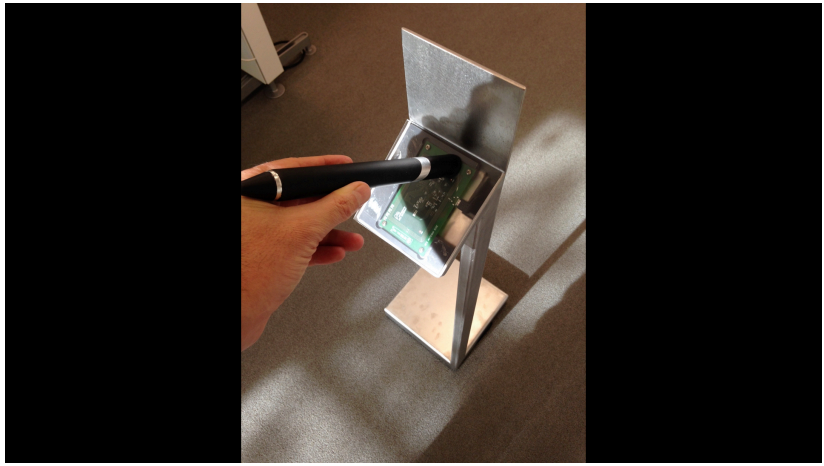
This is another one of the interactive tables in what is known as the "wallpaper room". Projectors mounted on the ceiling allow the museum to display wallpaper and other wall-coverings from its collection as if in real life, namely at the scale of an entire room. Normally the Cooper Hewitt only collects smaller samples of wallpaper but even if the museum had acquired rolls and rolls of these designs it doesn't have enough physical space to show them, let along store them.

The wallpaper room is amazing and it's a good illustration of the kind of thing thing that would otherwise be impossible in a museum before the advent of digital technologies. The wallpaper room doesn't replace, or supersede the value of, the original swatches but it does do something those swatches can't do on their own.



The wallpaper room was also designed to allow visitors to draw their own patterns and project them on the walls. Visitors could them save these designs to their visit the same way they could collect objects in the galleries.

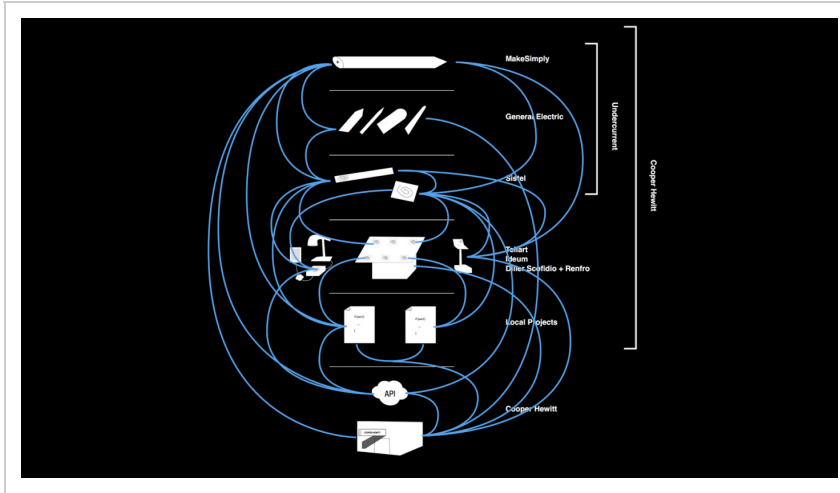
This is what the galleries looked like where every object listed on a wall label also had a target, the little cross in the top right-hand corner, with an embedded NFC tag containing that object's unique ID. Those IDs had a one-to-one mapping to the webpage for that object on the Cooper Hewitt **collection website** `https://collection.cooperhewitt.org/` .



This is a development photo of one of the "docking" stations for the Pen. The Pen was developed with a number of material and financial constraints which meant that although it could save objects a visitor collected it didn't have the means to send them anywhere. The Pen does not have wireless or internet capabilities and this meant that objects you collected had to be "dumped" using one of these stations or an interactive table. I'll come back to this in a minute.

This is a high-level breakdown of the different layers, or separation of concerns, required to design and produce hardware.

These are the principal actors responsible for each of those layers. The Cooper Hewitt did not build the Pen alone and it would not have happened except for the hard work and good will of these organizations.



This is an illustration of how all those organizations, and the individual components they were responsible for, interacted. In the end the Cooper Hewitt ended up being its own "general contractor" for this project. It was, as I said, a very very hard project.

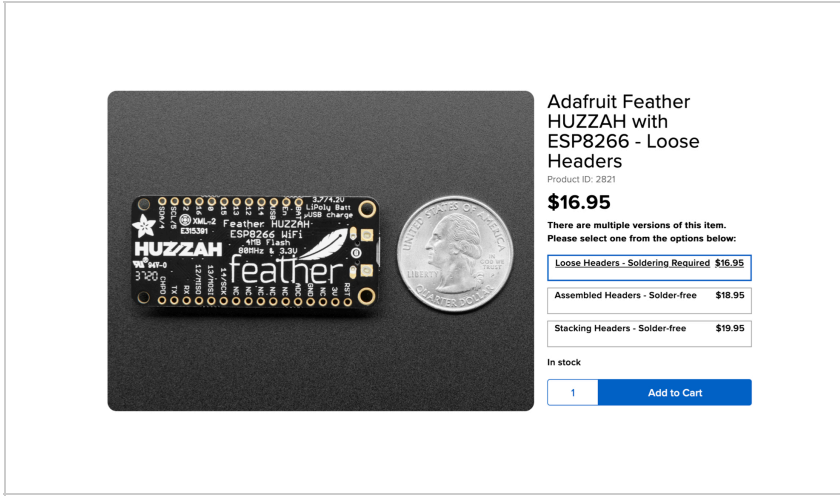
There are many things about the way we got the Pen from concept to finished product that I would not recommend. The Cooper Hewitt was operating within a set of circumstances which dictated some of the choices we made and how they were implemented but there are better ways to accomplish the same results.

One thing we did right, however, was to explicitly retain the intellectual property rights on key components of the Pen. We retained the rights the physical design and the interaction design of the Pen, originally developed by General Electric. We also retained the rights to the software used in the interactive tables, developed by **Local Projects** `https://localprojects.com/` .

These were very deliberate choices made to foster conditions, inside the museum, where the problems these tools were meant to solve might be revisited.

What I mean by that is that we, those of us working to bring the Pen to life, always saw it as a first iteration. It was important to try and build things in such a way that an iterative process to refine and improve the Pen was feasible within the economic and scheduling demands of the museum.

It's worth pausing here to acknowlegde that what I've just described, not only developing and launching a major initiative like the Pen but then continuing to revisit and potentially alter that work, rarely happens in the cultural heritage sector. I want to acknowlegde that only to acknowledge that it was a motivating decision that, with hindsight, we did a poor job of conveying to others and was the source of some anxiety after the Pen launched.



This is a 2022 screenshot for the **Adafruit Feather HUZZAH** `https://www.adafruit.com/product/2821` microcontroller. This is a microcontroller with 4MB of RAM and a built-in wireless chip that costs less that 17$. It is as you can see about as wide as a quarter. I think that's pretty amazing.

This microcontroller is still not suitable as a drop-in replacement for the circuit boards that were developed for the Pen in 2014; it would need to be about two-thirds as wide and be powered by triple-A batteries. On the other hand if I were developing, or iterating on, the Pen today **Adafruit** `https://adafruit.com/` would be the first people I would call to see if we could work out the economics of those challenges.

This circuitboard is orders of magnitude more powerful and, importantly, cheaper than the circuitboard the Pen launched with. We understood this when we were designing the Pen and we always

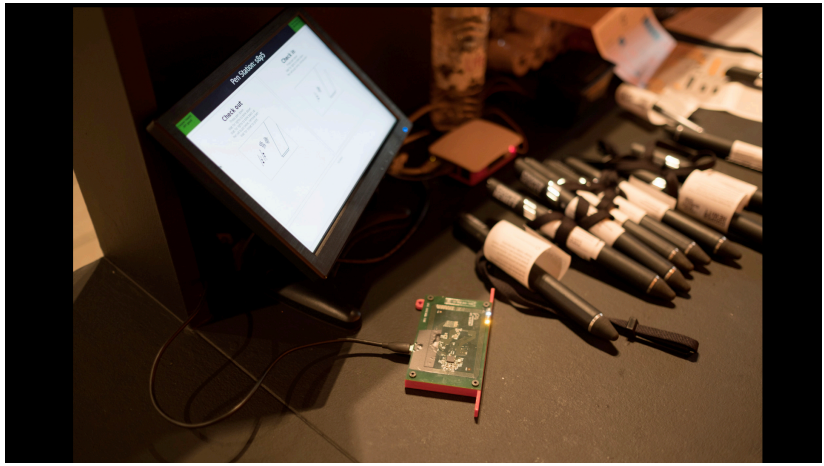
imagined that some day we could replace the original circuitboards with newer improved boards without having the replace the Pen itself. The Pen would continue to work as it always had in the galleries except that "version two", so to speak, might have a connection to the internet allowing objects collected to be associated with a person's visit automatically without the need to use a docking station or an interactive table.

Do you remember the picture of the docker station from earlier? Those were only necessary to account for the realities of developing hardware in 2014. The goal was always to remove those stations as network-enable electronics got smaller and cheaper, allowing us to improve the Pen rather than replacing it wholesale.

Adafruit has published **a really great history of the Feather boards** `https://learn.adafruit.com/adafruit-feather?view=all` , describing how it was first created and how it has continued to evolve since then. There are lessons here that I think all museums could learn from.

In the immediate years after the Pen launched Cooper Hewitt had success iterating on and improving the digital infrastructure it had built for the museum's reopening. In 2016 **it traveled both the Pen and the interactive tables to the London Biennale** `https://labs.cooperhewitt.org/2016/traveling-our-technology-to-the-u-k/` . In order to do this the digital team had to retrofit code it had written internally as well as code that Local Projects had written. The team created an equivalent "white box" implementation of nearly all the interactive experiences in the Cooper Hewitt's own galleries which they could take to London.

The team took a complex hardware and software ecosystem that was purpose-built for the Cooper Hewitt and its technology infrastructure and adapted it for use in an entirely different environment. They did all of this work in-house and on staff time. When I talked about "creating conditions where problems may be revisited" this is the kind of example I am talking about.



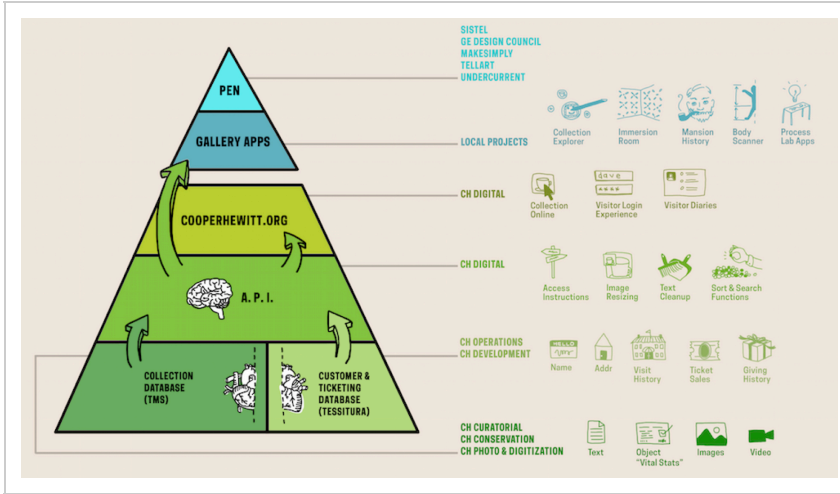
As part of traveling the Pen to London many of the interfaces, both public-facing and staff-specific, were revisited. **Entirely new tools for visitor services** `https://revdancatt.com/weeknotes/2020/06/20/cooper-hewitt-pens` to hand out pens and to associate them with tickets were written by a third-party developer on top of **Cooper Hewitt's programtic interfaces** `https://labs.cooperhewitt.org/2014/the-api-at-the-center-of-the-museum/` for the Pen-related activities.

The digital team at Cooper Hewitt started to experiment with **new interactive installations that wouldn't have been possible before the Pen** `https://labs.cooperhewitt.org/2016/process-lab-citizen-designer-digital-interactive-design-case-study/` . I want to stress, again, that just as important as the work of imagining and creating new interfaces was the fact that this work could be accomplished on staff time and within the boundaries of people's already busy schedules. Projects like this one did not require a new round of capital funding, or requests for proposals or the protracted timelines of so many vendor-driven solutions.

Before I left the museum in 2015 we also started experimenting with **allowing visitors to collect items for sale in the shop** `https://labs.cooperhewitt.org/2015/collect-all-the-things/` . It seemed only logical that at a design museum where many of the objects in the exhibitions were also available for sale, for example **this chair by Thomas Heatherwick** `https://collection.cooperhewitt.org/objects/35460799/` , that the same interaction affordances be available. We also started letting people collect the short essays on the wall that accompany exhibitions and had plans to do the same for individual architectural elements in the museum building. We even talked about letting people collect the plants and flowers on display in the museum garden.

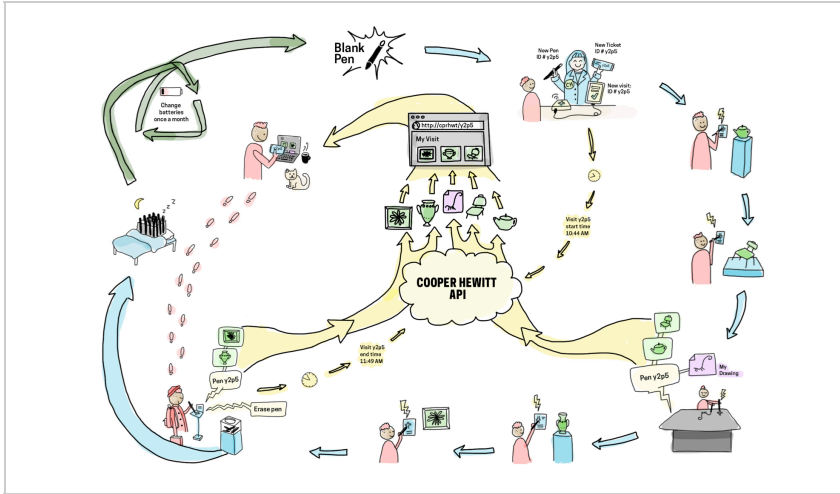
Some of these ideas were better, or more successful, than others. The important point is that they were all made possible, faster and cheaper, by the Pen. Without the Pen and the infrastructure to support it many of these ideas would have been too risky, too time-

consuming or too expensive to pursue. The Pen not only made them possible but made them possible in a way that allowed some of the more experimental projects to be worth investigating even if they fell short in practice.



This is a diagram, made by **Katie Shelley** `https://katieshelly.art` , illustrating how all the digital systems at the Cooper Hewitt fit together. So far, nearly everything I've been talking about has been the smaller triangle at the very top of the larger pyramid. The museums collection and the database of people who have some involvement with the museum (visitors, donors, patrons) are the base of the triangle; the foundation on which everything else rests. Just above that sits "the API" which was the short-hand we used to describe the layer of software and services that handled the basic operations around storage and retrieval for objects in the collection and the individuals who interacted with them.

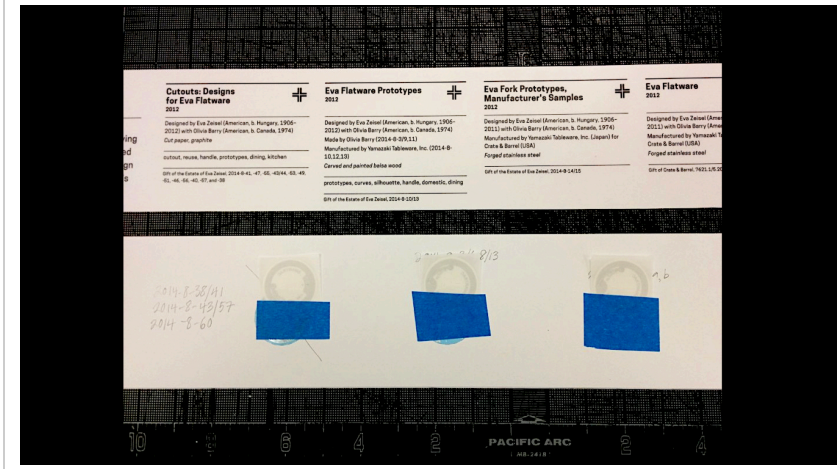
The first layer of the pyramid is what makes everything possible. The second layer, the "API", is the means by which that possible is accomplished. Everything else is a so-called "higher order process".



Here's another illustration that Katie made to visualize how everything was meant to fit together. In 2015 this was, in fact, the minimum number of steps required to implement what was effectively a physical bookmarking system in a museum.

In 2022 it is possible to imagine removing at least one step – having to retrieve objects from a Pen at the end of a visit, as shown in the bottom left-hand corner – by adding wireless network connectivity to the Pen itself. It hasn't happened and I don't think it ever will but this was always the plan.

I started out by talking about expectations and the idea that sometimes expectations "happen to you". Sometimes other people's expectations arrive unannounced and uninvited and suddenly they are a part of your daily life. This was certainly the case with the Pen at Cooper Hewitt. In order for the Pen, and all its associated interactions, to function **every department at the museum had to make changes** `http://www.aaronland.info/weblog/2015/04/10/things/#mw2015` to the way they worked.



Curatorial had to choose, and write extra descriptions for, related objects (to those on exhibition) to be included on the interactive tables. Registration had to make sure the collection management systems were current and up to date in advance of an exhibition opening. Visitor services had to intergrate configuring and handing out Pens when admitting visitors; they also had to deal with returning the Pens, trouble-shooting the Pens and keeping track of when the batteries in the Pens needed to be replaced. Graphic design and the installation teams had to account for the addition and placement of NFC tags on every single wall label. Security had to make sure people didn't steal the Pens.

I mention that because I think its important to recognize that we, the team responsible for the Pen, could have done a better job in explaining and helping people manage the changes the Pen was going to produce. We tried, as best we could, but we could have done more. The challenge with a project like the Pen is in trying to create belief in something that may not exist yet.

Perhaps one of the things we should have done was to have produced an equivalent, internally-facing record to mirror the public-facing weblog we published from 2012 to 2016. Nearly everything I've talked about today was written about, in even more detail, as it is being built. We put a high premium on working in public and of discussing the motivating factors behind our choices.

I think we imagined that staff would read the **labs.cooperhewitt.org** `https://labs.cooperhewitt.org` weblog along with everyone else but that never happened, at least not the way we thought it would.

We were asked, many times, why we didn't just build an "app" to do everything the Pen did. The really short answer is that it wasn't until five years after the Pen launched that reading NFC tags became common place on iPhones. There were some other reasons to but even if we'd wanted to make an app it would not have been possible given the number of visitors who own iPhones.

Android phones, on the other hand, have been able to read NFC tags since 2012.

It was our intention, sometime after the launch of the Pen, to make the same API methods that the Pen infrastructure used to collect and retrieve objects associated with a visit available to the general public. We want to do this specifically to make it possible for external developers to write Pen-like applications for people's mobile devices. One of the many lessons learned with the Pen is that we should have done this sooner. We didn't and then a whole bunch of organizational intertia and other dynamics began to press on the Pen

and those APIs were never opened up.

Fast forward to 2020 when NFC readers, and their corresponding APIs, finally became available in low to mid-range iPhones. I got my hands on one of those phones and decided to see **what a phone-based Pen equivalent would look like** `https://aaronland.info/weblog/2020/06/16/revisiting/#pen` . Here's a video demonstrating the basic interactions.

One of the things this experiment showed me is that it's not at all obvious where the NFC reader is located on a phone and that it needs to be placed in a very deliberate manner over an NFC tag. At Cooper Hewitt we spent a lot of time thinking about how to make touching a wall label with a Pen an action that, once learned, you wouldn't notice. We spent a lot of time thinking about how to minimize the amount that a person's wrist would need to bend in order to position the Pen in a way that it could read an NFC tag, many of which were mounted flat on the wall rather than angled on a rail.

If a person is forced to repeatedly bend their wrists it very quickly becomes distracting and eventually painful. I am reasonable confident that we would not have been able to control any of these issues with an app-based Pen device and that the uptake and usage would have been pretty low as a result.

Eventually I expanded that Pen-app experiment to **scan wall labels from other collections** https://aaronland.info/weblog/2020/07/13/experience/#remembered . Here's me collecting a mocked-up **William the hippopotamus** https://www.metmuseum.org/art/collection/search/544227 from the Metropolitan Museum of Art's collection. In the previous example, the application was fetching data about an object from the Cooper Hewitt's API. In this example the application is fetching data from a local database of information derived from **the Met's OpenAccess collection data** https://github.com/metmuseum/openaccess/ , stored on the phone.

Android phones have always outpaced their iPhone-equivalents when it commes to publicly available NFC or RFID features. For example it is possible to write an Android application that **makes a phone behave as though it were an NFC tag** https://en.wikipedia.org/wiki/Host_card_emulation . That means it's possible to write an application that re-assigns the value of the NFC tag it publishes dynamically using data stored locally or over the network. It means that any manner in which an Android phone is able to receive or "sense" data can be used to automatically update the "NFC tag" it is pretending to be.

To illustrate this idea I wrote the dumbest application ever called **NFC Clock** https://millsfield.sfomuseum.org/blog/2020/11/09/nfcclock/ . As you might guess from the name the application updates the value of an NFC tag that it is "broadcasting" every second with the current date and time.

The reason any of this is interesting to me is that this Android phone cost a little more than two hundred dollars. That's probably still too expensive to make it practical to replace traditional wall labels, with or without embedded NFC tags as was the case at Cooper Hewitt, but when you think about the time and cost required to produce them (traditional wall labels) it feels like something that will become affordable sooner than we image.

The phones shown in the video have been placed one on top of the other to account for the very time-consuming challenges of getting any two consumer-grade devices to line up their respective NFC

components so that they might successfully communicate with one another. That's just one of number of outstanding issues for this work not to mention the challenge of keeping all the batteries in those all those phones charged. But it points in the direction of what could be possible, as electronics continue to be miniaturized and commoditized. In the same way that the Pen was meant to serve as a springboard for future, as-yet unimagined, interactions so too might a low-cost and network-enabled wall label with excess computing power and a screen.

An interesting wrinkle in all of this, for me, is that none of these examples would work at SFO Museum. Nearly all of the object-specific signage for our exhibitions is six inches behind glass and often at an angle. That's too far away for NFC-based technologies, whether it's the Pen or an app, to work.

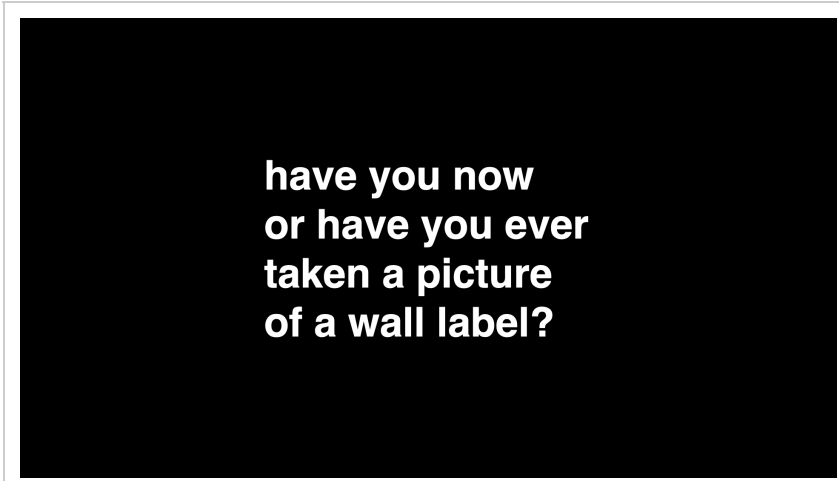
The other wireless technology you hear a lot about, especially in museum circles, is Bluetooth. Specifically low-energy Bluetooth beacons which do little more than broadcast a value, that has been assigned by its owner, to nearby Bluetooth receivers. These beacons are designed to use as little power as possible allowing them stay charged for months, sometimes years.

Bluetooth beacons have been notoriously complicated to set up and maintain, rarely as reliable as they claim to be and can quickly become prohibitively expensive. But it is also the case that mobile phones can act as de-facto beacons. Building off the experience of the NFC Clock application I modified the existing Pen-like mobile application to optionally scan for Bluetooth messages, in this case from the phone sitting next to it, in addition to NFC tags.

This is an imperfect solution for collecting individual objects but, in the case of say SFO Museum, it could be enough to allow people to collect an entire exhibition. Or perhaps we might install a tablet-sized device alongside an exhibition that displayed a carousel of images of each object on display broadcasting that object's ID along the way.

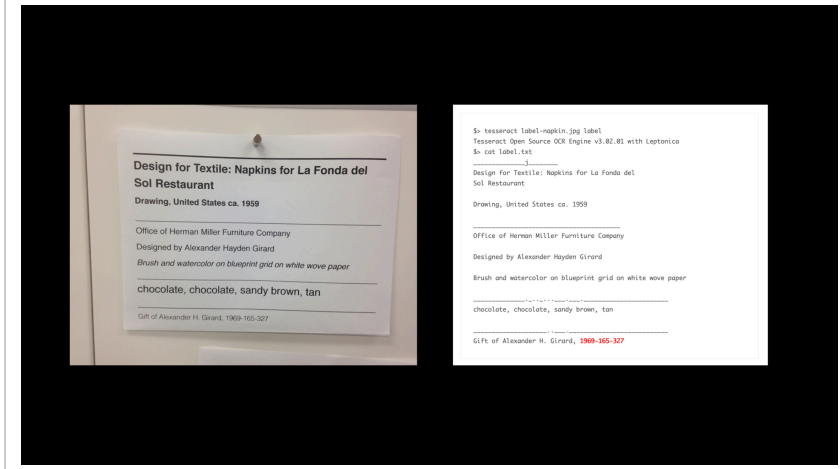
Are there perhaps better, more intuitive ways, for passengers to collect a whole exhibition? Absolutely. The point of this application was to try and understand the state of commercially available technologies that could serve as a way for the museum's activities to be present and participate with a network of devices belonging to passengers traveling through the airport.

What are the opportunities that those conditions afford the museum? What are the expectations that those conditions would prompt from passengers?



About once a month, during the time that Pen was being imagined, the digital team at Cooper Hewitt would spend a morning asking ourselves: What will we do if the Pen doesn't work? This is a good exercise for any project. Given the challenges of manufacturing custom electronic hardware it was always possible the Pen would fail but, regardless of that outcome, the museum would still re-open. If the purpose of the Pen was to allow people to remember their visits how would we do that without the Pen?

We used to say that the Pen would allow people to stop taking pictures of wall labels in museums. Taking pictures of wall labels was, and still is, the original Pen-like activity: A way to record something you've seen for future recall. The problem with those pictures is that they exist in a vacuum. Once recorded they are simply pixels in a larger catalog of pixels with no good way to organize or query any of the data contained in that picture.

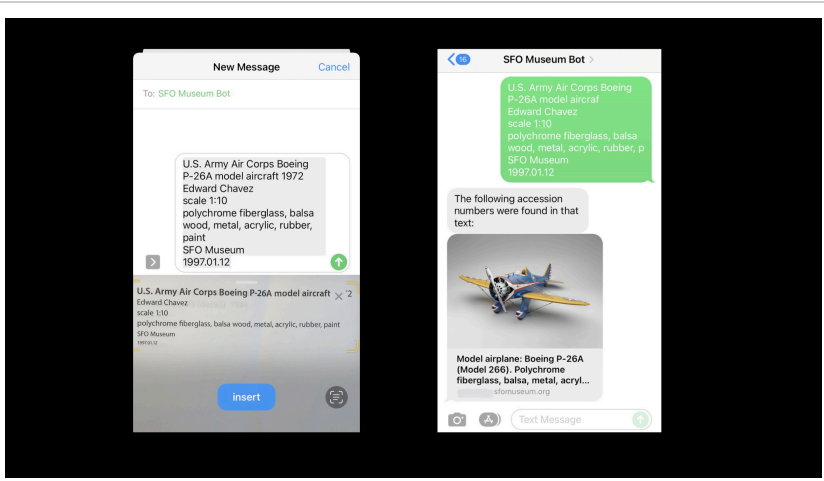


What if there were a way to extract the meaning, or at the very least, an identifier that pointed to meaning from a picture of a wall label? What if you could upload a photo to a server and then use optical character recognition software and pattern matching to pick out an accession number from the text in a label? If that were possible that same server could query a database for information about that object and then use that information to perform a higher-order process: Save the object to a visitor's account, display an essay about that object, and so on.

The idea was that a visitor could take action, in this case photographing a wall label, that when paired with a digital infrastructure operated by the museum would produce something more useful than the product of the action alone.

We built a prototype application called **Label Whisperer** `https://labs.cooperhewitt.org/2014/label-`

`whisperer/` that worked in ideal conditions where "ideal conditions" meant photos were taken in good lighting with label text centered and not at an angle. In practice, photos taken during a museum visit happen in poor to low lighting conditions and at funny angles because of the placement of labels or the number of people crowding them. In 2014 we could see the outlines of what might possible but the technological means were not available to either the museum or its visitors.



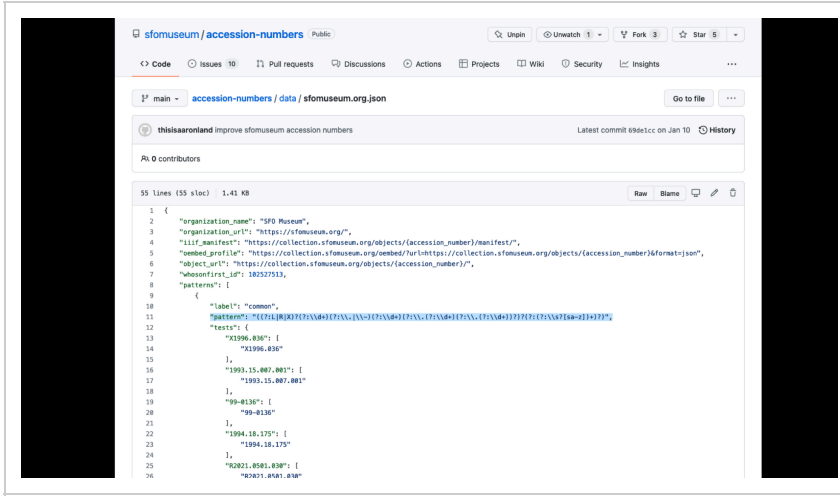
Fast forward, again, to 2021 and the nearly all Android and iOS devices now ship with the ability to recognize and extract text from still images. In the last year I've been **revisiting the ideas behind the Label Whisperer project** `https://millsfield.sfomuseum.org/blog/2021/12/30/accession-numbers/` with an eye to how they might be put to use at SFO Museum.

These are some screenshots of **a simple text messaging application** `https://github.com/sfomuseum/go-accession-numbers/#twilio-handler` that allows a visitor to photograph a wall label and then automatically insert the text of that label in to a text message. The message is then sent to a phone number we operate which triggers an application to parse the text for accession numbers, find the corresponding object in our collection and send a return (text) message containing a URL to that object on our collection website. We've included enough descriptive metadata, including an image, in **the webpage for that**

object `https://collection.sfomuseum.org/objects/1511907557/` that the phone's message application can display a preview of the object a person has just scanned.

That's all the application does so far but it's easy to imagine more sophisticated functionality built on to of it. The crucial point is that we've made identifying and retrieving the digital record for an object easy, or at least easier than it's been in the past.

Basically any and every digital "experience" you can imagine involving a museum collection depends on being able to recognize and distinguish individual objects. Nothing else works unless and until there is a shared understanding of what a person is looking at. The problem is relatively easy to state. The solutions to that problem are harder, in practice, than most people imagine. Hard does not mean impossible though. If we are available to find a common practice to address this problem, that recognizes the technological and budgetary limitations of the cultural heritage sector, then it opens up a whole new set of possibilities for our collections both individually and, no pun intended, collectively.



The text messaging application I just showed could potentially work in any museum, or at least any museums that include accession numbers on their wall labels. The same basic interactions could apply to any museum and the only difference, from museum to museum, would be the logic to recognize and extract accession numbers from a body of text.

This idea led to the creation of **the Accession Numbers project** `https://github.com/sfomuseum/accession-numbers` . It is a catalog of machine-readable documents, one per museum, that describe the patterns used to identify an accession number. Accession numbers are a rich tapestry, so to speak, with little or no consistency internally let alone between institutions.

These documents are pretty dry if you don't spend time programming computers or thinking about regular expressions (patterns). These aren't documents meant for leisurely contemplation. They are the

instruments necessary to make computers perform a specific task (extracting accession numbers) in the service of a higher-order process.

To date there are a few dozen entries and it's been encouraging to see staff at other museums submit definition files for their own institutions. Some day I hope the project will contain definitions for every museum in the world.

Here's an example of an early-stage application that consumes those data files and allows a user to scan label text from *different* museums and then collect those objects (or more specifically the identifier for an object) to a local database.

This is the Pen but instead of reading NFC tags it uses computer vision and text recognition software to identify an object from a wall label. That means any museum could offer the same Pen-like experience or collecting objects without the need for custom software or expensive and bespoke hardware or, even, the need to change the way that wall labels are produced or installed.

That means, potentially, all museums could offer their visitors the same ability to collect, to remember and revisit, objects in their collections with little to no additional cost.

At least in principle. First of all not all museums include accession numbers on their wall labels. That's easy enough to fix, though. The larger problem is that many, if not most, museums don't provide any way to display a standalone webpage for an object using an accession number. Most museum websites let you search for an object using an accession number but then render a list view where an object might only be one of many results displayed.

Sometimes museums don't, or won't, have webpages for objects they display in their galleries.

This is the case at SFO Museum. This is the case at SFO Museum for majority of objects that, historically, we have displayed in the terminals. These are exhibitions made up of objects on loan from other museums or donors and SFO Museum takes the position that it should not publish references to those objects on any of its websites because that might suggest they are part of our collection. This holds true even for web pages that may not even contain images because of

licensing or copyright reasons, but that would only display the same information we already print on our wall labels (and which people are taking pictures of).

I have not been able to convince the museum we should change that position, at least not yet.

I have been spending a lot of time talking about the importance and the need for creating stable and permanent references for collection objects. And when I say "references" what I mean are web pages with URLs (identifiers) that don't change. Web pages that can be viewed by humans and that have enough structure so that they can be also interpreted by machines and computer programs.

These web pages may not, in and of themselves, be terribly interesting. Their value comes from being, in very real and practical terms, the means by which more people than has even been possible can operate and participate in an expanding network of web pages. This network of documents can, in turn, function as a kind shared, collective storage and retrieval system which can be used to produce higher order processes. The web is thirty years old now and we largely take it for granted but I think it is important that we remember what a big deal it was when Tim Berners Lee first shared it with the world. There are two notable aspects about the web that distinguish it:

The first is that it found a sweet-spot between the functionality it offered and the skill necessary to participate, whether that is writing a web page or writing a web server or even a web browser. Web technologies have gotten exponentially more sophisticated and complex over the years but the basics – a network of linkable documents with degrees of graphic design elements – remains simple enough that implementing them doesn't need to seem like an impossibility. The second is that the web was released unencumbered from any licensing. If it's hard to appreciate why these things are important consider two constrasts, one from the recent past and one from the near future.

In the past, before the web, the principal means that a museum had to share its collection beyond the confines of its galleries were books. Those books were expensive to produce and to store and their reach was always limited by the economic and editorial constraints of publishers and distributors. In the near future, we are told, the web will be replaced by some form of virtual, mixed or augmented reality.

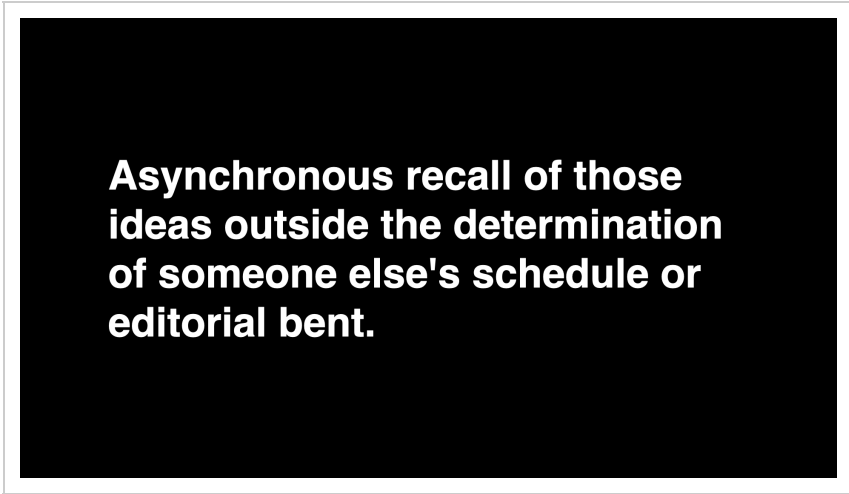
The possibilities of these technologies are fascinating to be sure but it is important to recognize that they depend on infrastructures and operational capabilities which only a handful of companies are able to afford and sustain. These are for-profit companies whose decisions are guided by economic and share-holder concerns which means that in the final analysis they will always remain walled gardens. If we adopt these new technologies uncritically we will have, effectively, replaced the web with a fancier and more sophisticated system of economic and editorial chokepoints that came to define the systems which pre-dated the web.

The web is the means by which the cultural heritage sector might pursue its mission not entirely divorced from economic constaints but divorced from economic constraints imposed on it by external actors. The web embodies principles of openness and portability and access that best align with the needs, and frankly the purpose, of the cultural heritage sector. The web is the instrument that makes these things possible in ways that did not exist before the web and we should make an effort to remember, cherish and nuture that fact, even if it is 30 years old.

The ability to project an idea beyond the range of any one individual's voice, beyond the limitations imposed by physical space and other geographies.

The web provides the ability to project an idea beyond the range of any ones individual's voice, beyond the limitations imposed by physical geographies. That's a big deal. That is an ability that, before the web, was reserved to a minority of people. The ability for an individual or a group to publish something that is within the reach everyone else on the rest of the planet is genuinely novel.

In recent years I think we have confused the value of simply being able to share ideas more broadly with the immediacy with which those ideas are, or are not, rewarded. There is a belief that if something isn't immediately successful, or more specifically *popular*, then it was a waste of time and a failure.



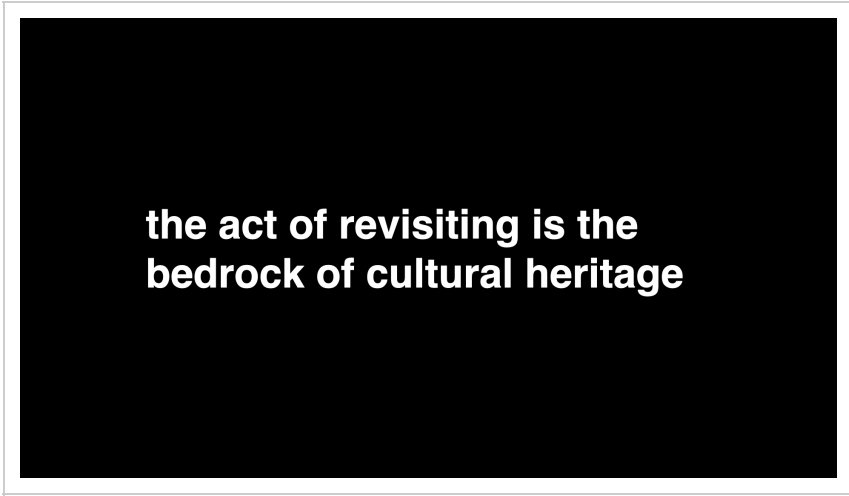
I think that's a terrible argument on first principles since it ignores the reality that lots of good ideas weren't understood or accepted in their time. History is littered with ideas that only upon reflection were understood to be valuable. It also deliberately ignores another distinguishing aspect of the web which is that web, by its design, makes it possible to find and retrieve an idea that someone has published days, weeks or years later and to have the means to do so independent of someone else's business model.

It has always been possible to gate access to stuff published on the web with time-based or economic restrictions but these are limitations which are layered on top of the web and not a fundamental property of how the web operates.

It has not always been possible to both store and retrieve the volume of stories and ideas, or even just lists of facts, that a global network of documents (web pages) enables accessible at a global scale. Or rather, if it was possible before the web it was only possible within

the boundaries of a vendor's proprietary and for-profit walled garden.

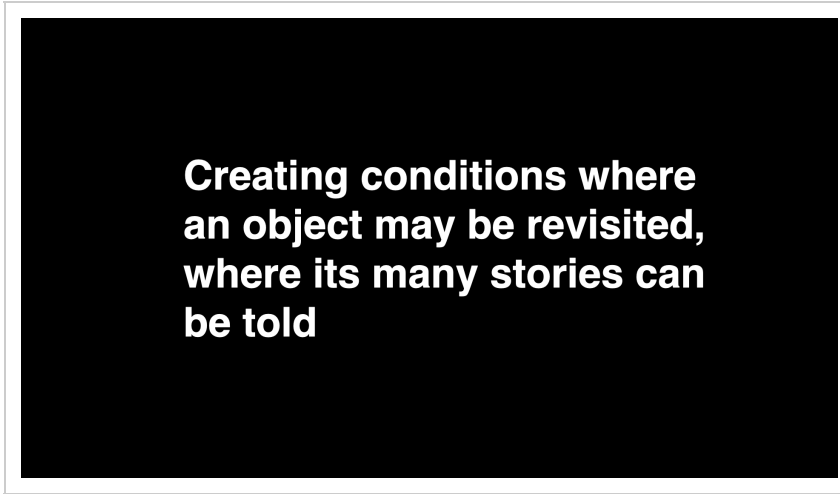
I believe that's an important consideration if the goal of the cultural heritage sector is to extend the reach and make accessible the things we deem important enough to preserve.



One of the reasons I think these considerations are so important to the cultural heritage sector is that the act of revisiting is the bedrock underlying the whole idea of cultural heritage itself. The act of re-reading a book, or re-listening to a piece of music or of going to see the same painting or sculpture again and again and of celebrating the evolution of an understanding about those works is what separates cultural heritage from entertainment. Entertainment, born of the moment, can and does become cultural heritage but it is precisely through the act of revisiting that one becomes the other.

The web simply makes those acts of revisiting possible for more people, in more ways, than ever before. When I say these things are "possible" I mean that the economics of participating in this global network of documents are within the reach of, if not everyone, then more people than ever before. These things become possible because technologies underlying the web are simple enough to meaningfully lower the barrier to paricipation and unburdened by licensing in a way which allows people to share their work and for others still to

extend that work to meet their needs.



I've talked a lot about the need to ensure that every object in a museum collection has a web page; that every object has a stable referent within a larger network of documents that can be reliably accessed, by humans and machines both, over time. I have tried to demonstrate that this is the foundational work which best enables the conditions, technologically and economically, for richer and more sophisticated "experiences" about our collections.

There is a legitimate critique of projects like the Pen which says that stable identifiers, persistent web pages and a catalog of bookmarks are not enough, that they do not offer a compelling narrative that conveys the meaning and nuance of a collection. I wholeheartedly agree with this argument but I think it depends on two interlocking issues which need to be addressed in order to do so.

The first is ensuring (often creating) the conditions, technologically and economically, which enable those stories to be told and, more importantly, to be retold in a variety of ways. Magaret MacMillan's

quote that "the past keeps changing because we keep asking different questions of it" bears repeating. The ability to ask new questions of our collections, without the need for another multi-million dollar, multi-year capital project, should be the measure of the technologies we adopt and deploy. I believe that embracing and promoting the web, an open and shared network of linked documents with stable references, is the best way to do that.

The second involves actually retelling, or revisiting, that work and using the objects to craft narratives which are greater than the sum of their parts. That hasn't really happened yet. It absolutely should. The fact that, collectively, the sector hasn't been able to do that work yet suggests to me that the sector actually has no experience doing the things we say we should be doing.

Exhibitions remain, by and large, "fire and forget" style endeavours. A lot of time and effort goes in to producing an exhibition but once that exhibition has launched it is, effectively, frozen in amber. Rarely, if ever, does a museum revisit or reconsider or even reference past exhibitions. To the extent that there is additional programming – lectures, tours, blog posts – about an exhibition they typically only happen during the window of time that an exhibition is on display. To the extent that an exhibition is remembered at all it is remembered in the form of a printed book.

This a kind of institutional muscle memory than I believe we should work to overcome, precisely so that we might better craft stories and meaning around our collections. If you accept my argument that the technologies our visitors now take for granted have forever changed their expectations of what is, or should be, possible then ultimately

we must do these things.

Tools like the Pen or initiatives like the Accession Numbers project, while interesting from a technological perspective, are not sufficient in and of themselves. Rather they are efforts to try and solve the mechnical problems of bridging the physical world of our collections and our buildings to the larger network of linked documents we call the web and to do so within the economic and staffing realities of the cultural heritage sector.

The point of these projects is to create conditions where both the projects themselves and the stories they are meant to produce might be revisited. I sometimes talk about the need for the cultural heritage sector to "design for patience" in the work it does. It is true that books and printed material remain the most durable means for an idea to outlast people's reluctance; to be revisited. I am not here to suggest we replace that practice but to point out we have at our disposal a way to supplement and extend it at a scale that previously was practically impossible if not also unimaginable.

If Piketty was able to craft his arguments about wealth and income inequality from 200 years of tax records, which let's agree are not compelling reading for most people, what might be created from our collective works of "cultural heritage" were they afforded not only the same degree of persistance but also retrievability? Given that the means to do so are within our reach now why wouldn't we embrace them?

2022-06-17

like a giant frozen fish cake palette

Coastline, NorCal (2022)

Coastline, NorCal (2022)

12 inches by 48 inches. Chiyogami and paper on wood.

2022-07-05

things I have written about elsewhere #20220801

The Case of the Missing (Istanbul) Airport

The Case of the Missing (Istanbul) Airport

This was **originally published on the SFO Museum Mills Field weblog** `https://millsfield.whosonfirst.org/blog/2022/08/01/airports/` , in August, 2022.

Postcard: globe, Pan American Airways, Miami. Paper, ink. Gift of Thomas G. Dragges, SFO Museum Collection. ***2015.166.1228***

On the surface this is a blog post documenting the steps to add a new record (an airport) to a catalog of geographic places (the **sfomuseum-data-whosonfirst** `https://github.com/sfomuseum-data/sfomuseum-data-whosonfirst` GitHub repository). Scratching the surface, though, it's really a blog post about how SFO Museum supplements and extends the **Who's On First** `https://whosonfirst.org/what` to meet the needs of our online efforts.

Much of what follows has been discussed in the **Who's on First at SFO Museum** `https://millsfield.sfomuseum.org/blog/2018/08/28/whosonfirst/` and **SFO Museum, Who's On First and Airports** `https://millsfield.sfomuseum.org/blog/2018/10/30/airports/` blog posts from 2018. To recap: SFO Museum publishes its

openly licensed data https://github.com/sfomuseum-data as Who's On First -style GeoJSON records. We do this because place is inherent to the airport (SFO) and to SFO Museum's **aviation collection** https://collection.sfomuseumg.org . Who's On First allows the museum to connect its data with a global database of places using a plain-text format (GeoJSON) that is universally supported by developer and GIS (geographic information system) tools alike.

We source data for places outside the airport campus directly from Who's On First and we author and manage data inside the SFO campus modeling these records as though they were Who's On First records. This allows the two datasets to "just work" together. It is possible to associate our efforts not just with the rest of the world but also with any other project using Who's On First to identify places and without the hassle of determining whether our specific records should be part of **the core Who's On First dataset** https://github.com/whosonfirst-data .

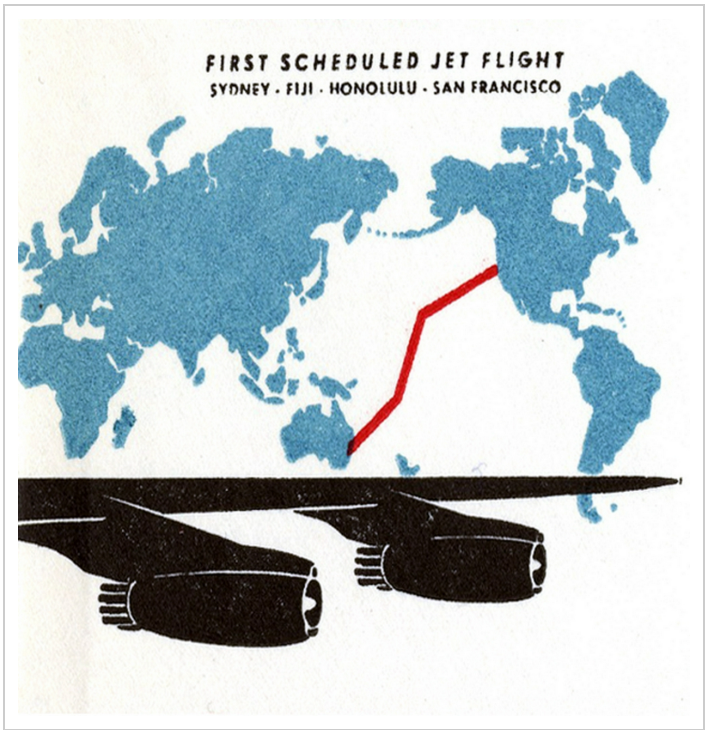
One class of places "outside" the SFO campus that we definitely care about are the other airports servicing **flights in and out of our airport** https://millsfield.sfomuseum.org/flights . You might think that if there was ever a class of places which already had canonical identifiers it would be airports. Without getting in to all the reasons why, except to point out that **three-letter IATA airport codes** https://en.wikipedia.org/wiki/International_Air_Transport_Association_code are periodically re-used, I'll just say that its often harder to distinguish one airport from another than it should be.

Model airplane: Turkish Airlines, Boeing 777-300ER. Plastic, paint, wood, metal. Transfer from City and County of San Francisco, Office of the Mayor, SFO Museum Collection. ***2017.073.001 a d***

To account for this problem we do a bunch of work behind the scenes to map airports, airlines and aircraft, regardless of how they choose to identify themselves, to their corresponding Who's On First record (and ID). That's fine so long as there is an existing Who's On First record for every airport... but what if there isn't? The number of new airports opening every year is pretty low but it's often greater than zero. This turned out to be the case with **Istanbul Airport** `https://en.wikipedia.org/wiki/Istanbul_Airport` which opened in 2018 and whose absence (in Who's On First data) we only noticed recently. Oops.

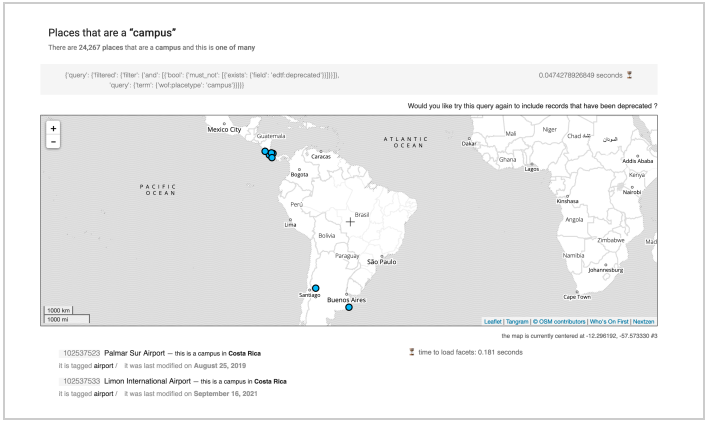
This is what we did to rectify that absence, told in two parts. The first part documents how SFO Museum imports existing Who's On First records in to its own data catalog. The second part documents what to do when there is a place SFO Museum wants to import but that doesn't have a record in Who's On First yet. These examples are, by and large, tailored to SFO Museum's needs but we are sharing them in the spirit of generosity to demonstrate how we manage a custom dataset of places on top of the "core" Who's On First data.

Adding a new airport to `sfomuseum/sfomuseum-data-whosonfirst`



Airmail flight cover: Qantas Empire Airways, Boeing 707. Paper, ink. Gift of Mrs. Siusiadh Rasmussen, SFO Museum Collection. ***2003.013.001***

As mentioned, **airport data** `https://spelunker.whosonfirst.org/placetypes/campus/` is sourced from the **whosonfirst-data** `https://github.com/whosonfirst-data` repositories stored on GitHub. The first thing to do, then, is figure out which respository a given airport record is stored in. You can use the **Who's On First Spelunker** `https://spelunker.whosonfirst.org/` to look up this data.



Note that some airports are stored in the ***whosonfirst-data-admin-xy*** `https://github.com/whosonfirst-data/whosonfirst-data-admin-xy` *repository because at the time of the "****great splitting of the whosonfirst-data-admin repository in to per-country repositories****"* `https://whosonfirst.org/blog/2019/05/09/changes/` *that airport's country of origin wasn't able to be determined. While importing airports that are in the* ***-admin-xy*** *repository is a good opportunity to move the record in to the correct repository that is not strictly necessary.*

Although data is sourced from the Who's On First (WOF) project, SFO Museum maintains local copies of WOF records in its own **sfomuseum-data-whosonfirst** `https://github.com/sfomuseum-data/sfomuseum-data-whosonfirst` repository. We do this for two reasons. First, we only need a subset of the total number of WOF records so it's easier to keep the records we care about in a dedicated repository. Second, we apply SFO Museum-specific properties to those records; changes which are not relevant or suitable to be merged in to the "core" WOF dataset.

The **go-sfomuseum-whosonfirst** `https://github.com/sfomuseum/go-sfomuseum-whosonfirst` package was written to provide tools for

importing data from WOF in to the `sfomuseum-data-whosonfirst` repository. For example:

```
$> cd /usr/local/sfomuseum/go-sfomuseum-whosonfirst

$> ./bin/import-feature \
        -reader-uri 'github://whosonfirst-data/whosonfirst-data-admin-{COUNTRY}' \
        {ID} {ID} {ID}
```

The `import-feature` tool performs the following tasks:

1. *It will retrieve the record for each (WOF) ID specified, as well any relevant ancestors for that ID (region, country).*

2. *For each ID fetched it will create a corresonding JSON file in the `sfomuseum-data-whosonfirst/properties` folder. These JSON files are meant to contain any additional SFO Museum-specific properties or property values that should be overwritten (for example `sfomuseum:placetype` or `wof:repo`).*

Once the records have been imported make sure to commit them to the `sfomuseum-data-whosonfirst` repository:

```
$> cd /usr/local/data/sfomuseum/sfomuseum-data-whosonfirst
$> git add {NEW FILES}
$> git commit -m "Add ..." {NEW FILES}
$> git push origin main
```

Commiting the changes is relevant to the **go-sfomuseum-airfield** `https://github.com/sfomuseum/go-sfomuseum-airfield` package which provides pre-compiled lookup tables for things related to the SFO airfield (airlines, aircraft, airports). By default these tables are built by fetching the `sfomuseum/sfomuseum-data-whosonfirst` repository from GitHub. For example:

```
$> cd /usr/local/sfomuseum/go-sfomuseum-airfield
$> make compile
$> git commit -m "recompile data" .
$> git push origin main
```

Commiting the changes (to the `go-sfomuseum-airfield` package) is

also relevant because a lot of other tools that use those lookup tables build them on the fly by fetching the serialized tables over the wire from GitHub; for example **flight data** `https://millsfield.sfomuseum.org/flights` . Committing these lookup tables to GitHub and then retrieving them over the wire allows us to update airfield data without involving the time-consuming process of updating every other package that uses `go-sfomuseum-airfield`.

Adding a new airport to `whosonfirst/whosonfirst-data-admin-*`

Poster: Pan American World Airways, Routes of the Flying Clipper Ships. Paper, ink. Gift of the Pan Am Association, SFO Museum Collection. ***2000.058.0206 a b***

Sometimes (not often) there are new airports which haven't been added the Who's On First (WOF) project yet. This is an example of how to create a basic record for such a record, in this case **Istanbul Airport** `https://spelunker.whosonfirst.org/id/1779770747/` in

Turkey. The first step is to clone the **whosonfirst-data-admin-tr** `https://github.com/whosonfirst-data/whosonfirst-data-admin-tr` repository:

```
$> git clone \
	--depth 1 \
	git@github.com:whosonfirst-data/whosonfirst-data-admin-tr.git \
	/usr/local/data/whosonfirst-data-admin-tr
```

The next step is to build a SQLite database, with relevant spatial tables, that we can use to perform "point-in-polygon" operations to determine the new airport's parent and ancestors. Use the tools in the **go-whosonfirst-sqlite-features-index** `https://github.com/whosonfirst/go-whosonfirst-sqlite-features-index` package to create this database:

```
$> cd /usr/local/whosonfirst/go-whosonfirst-sqlite-features-index

$> ./bin/sqlite-index-features \
	-all \
	-timings \
	-dsn /usr/local/data/whosonfirst-data-admin-tr.db \
	/usr/local/data/whosonfirst-data-admin-tr
```

For Turkey it takes about 3-4 minutes to create the `/usr/local/data/whosonfirst-data-admin-tr.db` database. These spatially-enabled databases were previously discussed in the **Reverse-Geocoding in Time at SFO Museum** `https://millsfield.sfomuseum.org/blog/2021/03/26/spatial/` blog post, published in 2021.

Once the new spatial database has been created you need to reference it when invoking the `wof-create` tool, which is part of the **whosonfirst/go-whosonfirst-exportify** `https://github.com/whosonfirst/go-whosonfirst-exportify` package. For example:

```
$> cd /usr/local/whosonfirst/go-whosonfirst-exportify

$> ./bin/wof-create \
	-writer-uri repo:///usr/local/data/whosonfirst-data-admin-tr \
	-resolve-hierarchy \
	-spatial-database-uri 'sqlite://?dsn=/usr/local/data/whosonfirst-data-admin-tr.db
	-geometry '{"type":"Point","coordinates":[28.727778,41.262222]}' \
	-string-property 'properties.wof:placetype=campus' \
	-string-property 'properties.wof:country=TR' \
	-string-property 'properties.wof:name=Istanbul Airport' \
```

```
        -string-property 'properties.wof:repo=whosonfirst-data-admin-tr' \
        -int-property 'properties.mz:is_current=1' \
        -string-property 'properties.edtf:inception=2018-10-29' \
        -string-property 'properties.edtf:cessation=..' \
        -string-property 'properties.src:geom=wikipedia'
```

This will create a new, and minimal, record for the Istanbul Airport which can then be updated by hand as necessary. For example, this preliminary record has been created with only a `Point` geometry for the airport rather than a polygon depicting the geometry of the airport's campus.

For testing and debugging purposes you can emit the new record to STDOUT but assigning the `-writer-uri` flag like this:

```
$> ./bin/wof-create \
        -writer-uri stdout:// \
        {OTHER OPTIONS}
```

Commit the **new record** `https://spelunker.whosonfirst.org/id/1779770747/` (in this case to the `sfomuseum-data-admin-tr` repository) and then import it in to the `sfomuseum-data-whosonfirst` repository as described above.

Properties

Timetable: Turkish Airlines, summer schedule. Paper, ink, metal. Gift of Thomas G. Dragges, SFO Museum Collection. ***2015.167.755***

One reason we clone "core" Who's On First records in to a **sfomuseum-data-whosonfirst** `https://github.com/sfomuseum-data/sfomuseum-data-whosonfirst` repository is that we append those records with a variety of SFO Museum-specific properties. The Who's On First project aims to publish online and machine-readable definitions for any and all properties defined across its entire corpus. Given a property in a Who's On First record it should be possible to easily derive the URL for its definition in the **whosonfirst-properties** `https://github.com/whosonfirst/whosonfirst-properties` respository. For example the URL for the `wof:placetype` property is:

1. ***https://raw.githubusercontent.com/whosonfirst/whosonfirst-properties/main/properties/wof/placetype.json*** `https://raw.githubusercontent.com/whosonfirst/whosonfirst-properties/main/properties/wof/placetype.json`

Where `wof:placetype` becomes `wof/placetype.json` and the corresponding definition file looks like this:


```
{
    "id": 1158807975,
    "name": "placetype",
    "prefix": "wof",
    "description": "Represents a common, common-optional, or optional string value of wha
    "type": "string"
}
```

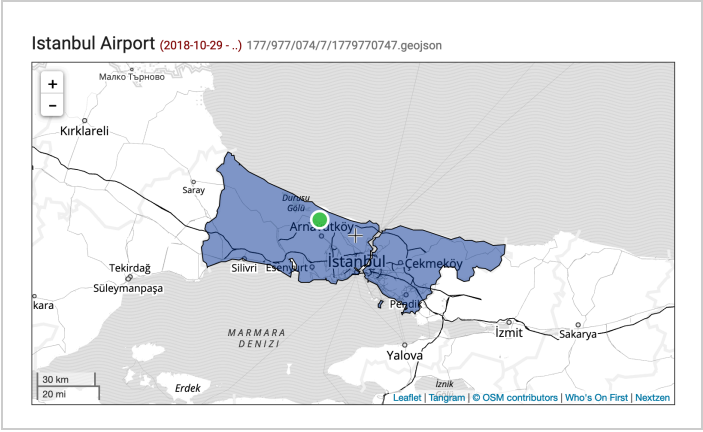

These property definition files build on **work that the Cooper Hewitt proposed** `https://labs.cooperhewitt.org/2012/proposal-glossaries-dot-json/` , in 2013, for cultural heritage organizations to publish a common "glossary.json" file describing the keys, labels and properties they use to describe their collections.

> It's long been something of a holy grail of museums, and cultural heritage institutions, to imagine that we can define a common set of metadata standards that we will all use and unlock a magic (pony) world of cross-institutional search and understanding. The shortest possible retort to this idea is: Yes, but no.
>
> We can (and should) try to standardize on those things that are common between institutions. However it is the differences – differences in responsibilities; in bias and interpretation; in technical infrastructure – that distinguish institutions from one another. One need look no further than the myriad ways in which museums encode their collection data in API responses to see this reality made manifest.
>
> I choose to believe that there are good and valid, and very specific, reasons why every institution does things a little differently and I don't want to change that. What I would like, however, is some guidance.

In the same way that SFO Museum layers its own data on top of Who's On First records we have created a separate collection of SFO Museum-specific property definition files stored in the **sfomuseum-properties** `https://github.com/sfomuseum/sfomuseum-properties` repository to supplement properties already described in the **whosonfirst-properties** `https://github.com/whosonfirst/go-whosonfirst-properties` repository. We manage things using the `index-properties` tool in the **go-whosonfirst-properties** `https://github.com/whosonfirst/go-whosonfirst-properties` package. For example, we can index all the properties in all the `sfomuseum-data-*` repositories storing only those not already found in the `whosonfirst-properties/properties` folder (and excluding properties starting with `misc:`) like this:

```
$> ./bin/index-properties \
        -exclude 'misc\:.*' \
        -alternate /usr/local/whosonfirst/whosonfirst-properties/properties \
        -properties /usr/local/sfomuseum/sfomuseum-properties \
        /usr/local/data/sfomuseum-data-*
```

As of this writing many of the SFO Museum property definition files are incomplete, lacking human readable descriptions. Those will follow in time. An incomplete record is not ideal but it is better than no record at all. An imcomplete record having announced its presence can be updated and improved. It is difficult to say the same of a record whose existence isn't recorded anywhere. Like, for instance, the no-longer missing record for **Istanbul Airport** `https://spelunker.whosonfirst.org/id/1779770747/` or all the records the **sfomuseum-data-flights-** `https://github.com/sfomuseum-data/?q=sfomuseum-data-flights-&type=all&language=&sort=` repositories that pointed to the wrong airport and which have **since been updated** `https://millsfield.sfomuseum.org/airports/IST/flights/` .



Software mentioned in this blog post

There's a lot of code working behind the scenes to make all of this possible. Here is the list of software packages explicitly referenced in this blog post:

1. ***sfomuseum/go-sfomuseum-whosonfirst*** `https://github.com/sfomuseum/go-sfomuseum-whosonfirst` *– Go package for working with whosonfirst-data repositories in a SFO Museum context.*

2. ***sfomuseum/go-sfomuseum-airfield*** `https://github.com/sfomuseum/go-sfomuseum-airfield` *– Go package for working with airfield-related activities at SFO Museum (airlines, aircraft, airports).*

3. ***whosonfirst/go-whosonfirst-sqlite-features-index*** `https://github.com/whosonfirst/go-whosonfirst-sqlite-features-index` *– Go package for indexing Who's On First features in SQLite databases using the* `whosonfirst/go-whosonfirst-sqlite-index` *and* `whosonfirst/go-whosonfirst-sqlite-features` *packages.*

4. ***whosonfirst/go-whosonfirst-exportify*** `https://github.com/whosonfirst/go-whosonfirst-exportify` *– Tools (written in Go) for exporting Who's On First (WOF) records.*

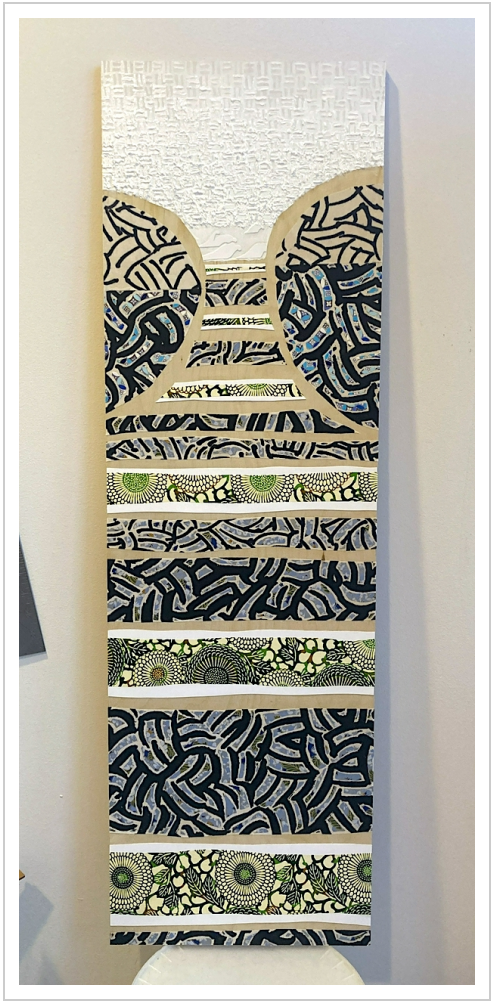
5. ***whosonfirst/go-whosonfirst-properties*** `https://github.com/whosonfirst/go-whosonfirst-properties` *- Go package for working with Who's On First properties.*

---

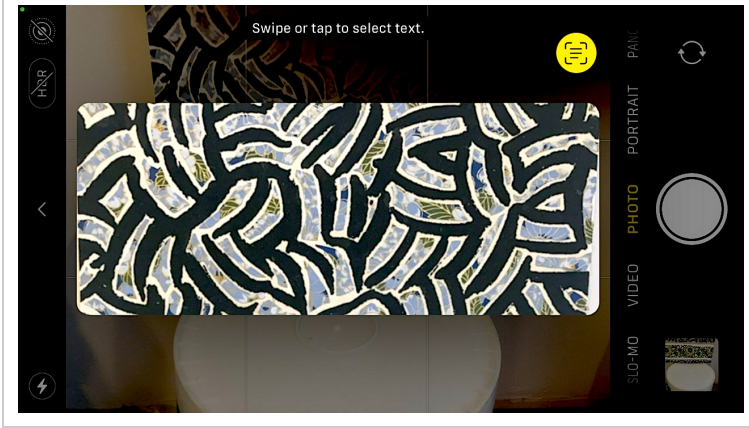
listening past the delivery

Estero Americano #2 (2022)

Estero Americano #2 (2022)



36 by 12 inches. Paper, chiyogami, fabric and oil on wood.
Meanwhile, if anyone can tell me what language my phone thinks it's seeing here I'd be much obliged.


Swipe or tap to select text.
HDR
PANO
PORTRAIT
PHOTO
VIDEO
SLO-MO

Update: As sometimes happens I wasn't quite finished with this one when I first posted this. These changes are not a "study of" or a "reference to" John Powers' **white on white paper drawings** `https://www.johnpowers.us/` but I always think of them when I do stuff like this so credit where credit is due.

John is a sculptor and thinks painters are from another planet so I don't know if he is aware of the generally accepted rule in painting which says you can layer oil on top of acrylic but not the other way around. I know that rule but went ahead and glued those little pieces of paper on with acrylic gel anyway. As such, I am resigned to the possibility that some amount of my future will be devoted to gluing

them back on as they fall off over the years.

2022-08-05

things I have written about elsewhere #20220808

Map updates, August 2022

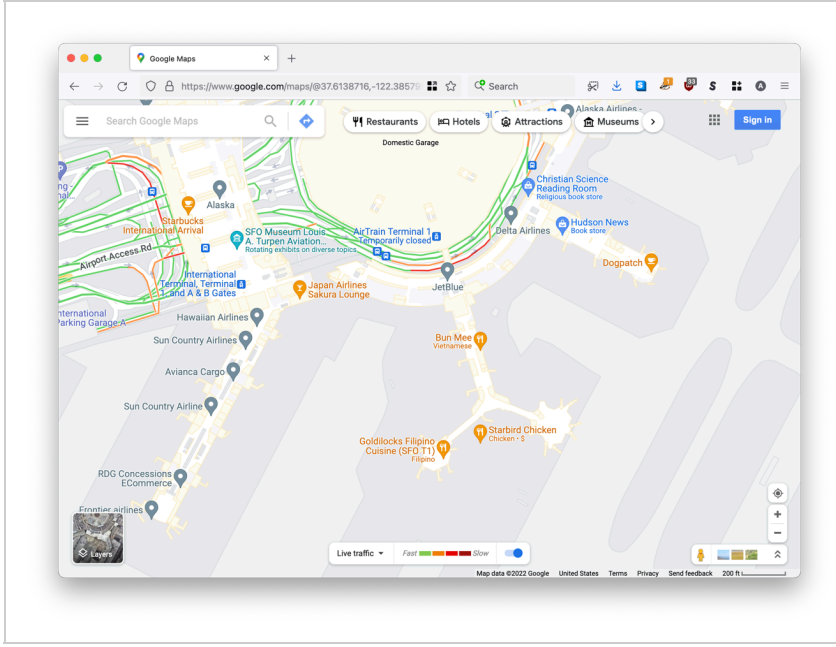
Map updates, August 2022

This was **originally published on the SFO Museum Mills Field weblog** https://millsfield.sfomuseum.org/blog/2022/08/08/maps/ , in August, 2022.

We've updated all our maps to include aerial imagery captured in July, 2022. This includes all the maps on the **Aviation Collection** https://collection.sfomuseum.org and **Mills Field** https://millsfield.sfomuseum.org websites as well as the interactive installation in the **Terminal 2 SkyTerrace** https://millsfield.sfomuseum.org/blog/2021/11/19/skyterrace/ .

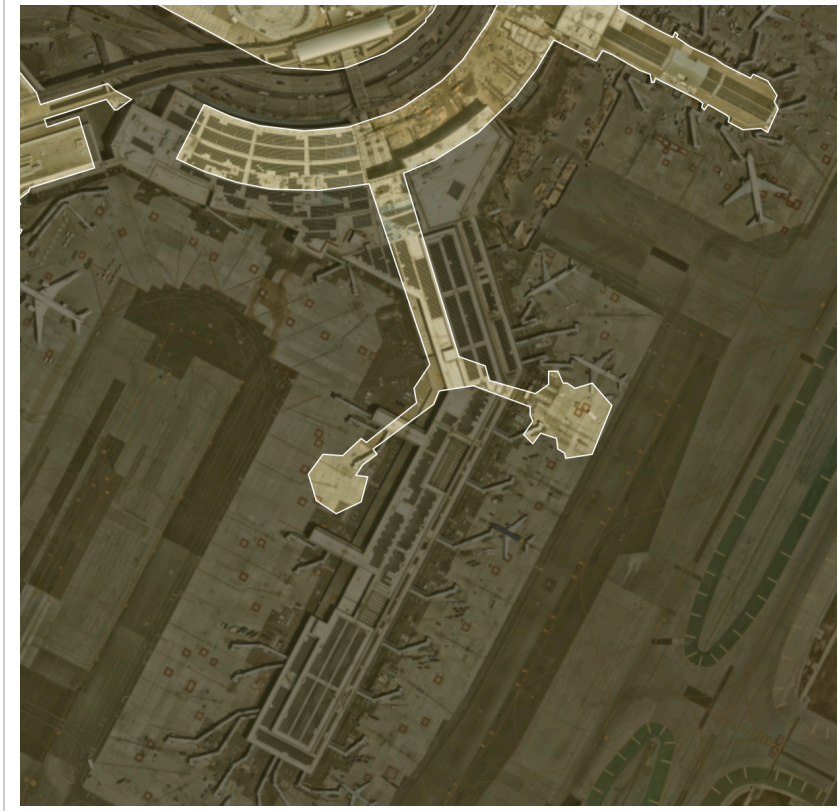
This availability of this new imagery was taken as an opportunity to refactor the code used to display maps across all our websites (and the SkyTerrace). This code has grown organically over the years to the point where a bit of corraling was in order. We gathered up all the different features across all these different maps and merged them in to a set of JavaScript libraries which provide common building blocks for generating maps. These include custom libraries for handling global base map tiles, historical aerial imagery, a handy control to resize the height of a map and, importantly, a stylized representation of the SFO campus and terminal complex derived from data provided by the airport's GIS department.

Here are a series of screenshots, all captured on August 02, 2022, to illustrate why that's important:

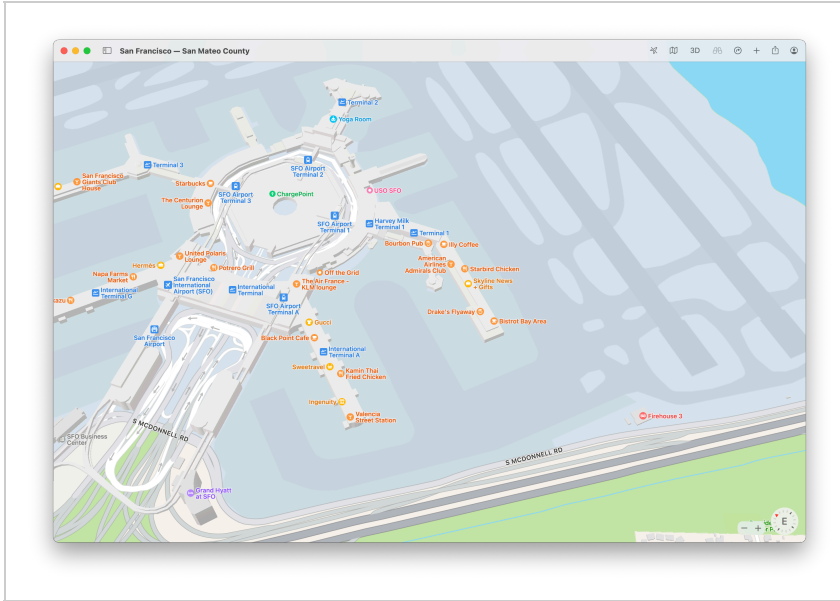


This is what **Google Maps** https://www.google.com/maps/@37.6136506,-122.3881341,17z/data=!5m1!1e1 thinks SFO looks like in 2022. SFO did look like this once, but that was during the years between **2014 and 2017** https://millsfield.sfomuseum.org/buildings/1159396321/ . Since then the **old Boarding Area B** https://millsfield.sfomuseum.org/boardingareas/1159396237/ with its two rotunda have been demolished and replaced by the new **Harvey Milk Terminal 1 boarding area** https://millsfield.sfomuseum.org/terminals/1477856657/ .

This is what the old Boarding Area B looks like overlaied on the new satellite imagery from July, 2022.

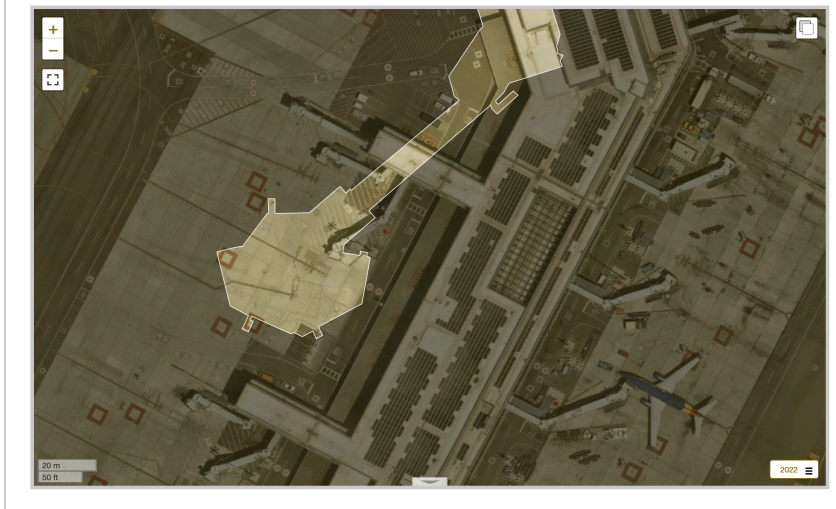


Apple Map's understanding of SFO looks to be dated to **sometime in 2019** `https://millsfield.sfomuseum.org/buildings/1477855605/` .

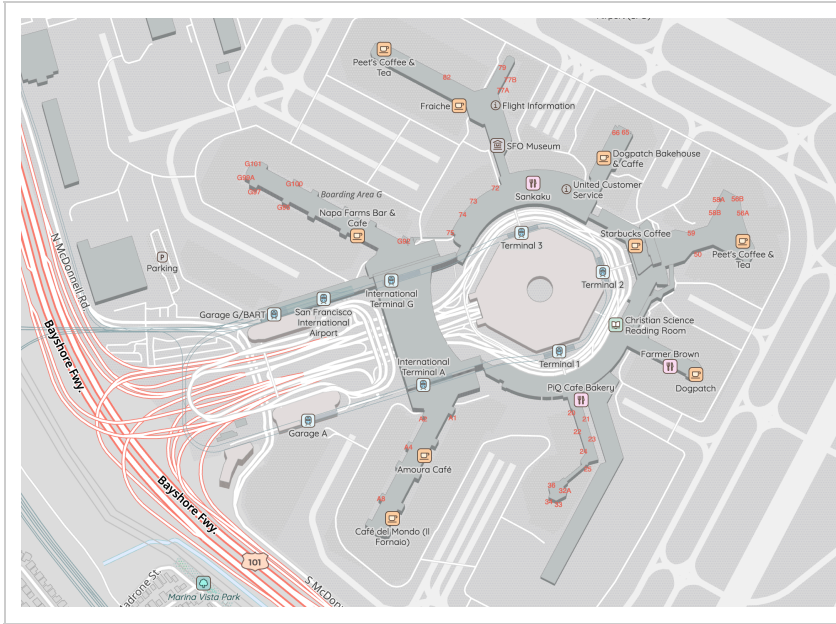


That would be around the time that the new Harvey Milk Terminal 1 building opened but before the last rotunda was demolished.

In 2022, SFO is basically parking planes in the space where that rotunda used to be.

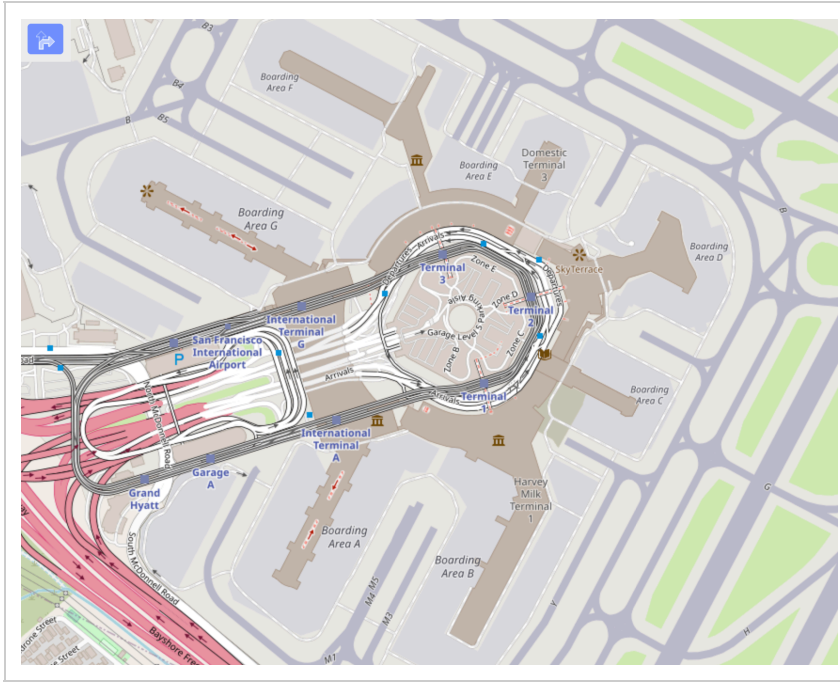


The same is even true of map tiles produced by the **Nextzen** `https://nextzen.org` project which we've used extensively for **our own maps** `https://millsfield.sfomuseum.org/blog/tags/rasterzen` . Like Apple, Nextzen thinks SFO is stuck in 2019. This is evident not just from the old architecture but also the gate numbers which **were renumbered in October, 2019** `https://www.youtube.com/watch?v=-tTzI8ufeQA` .

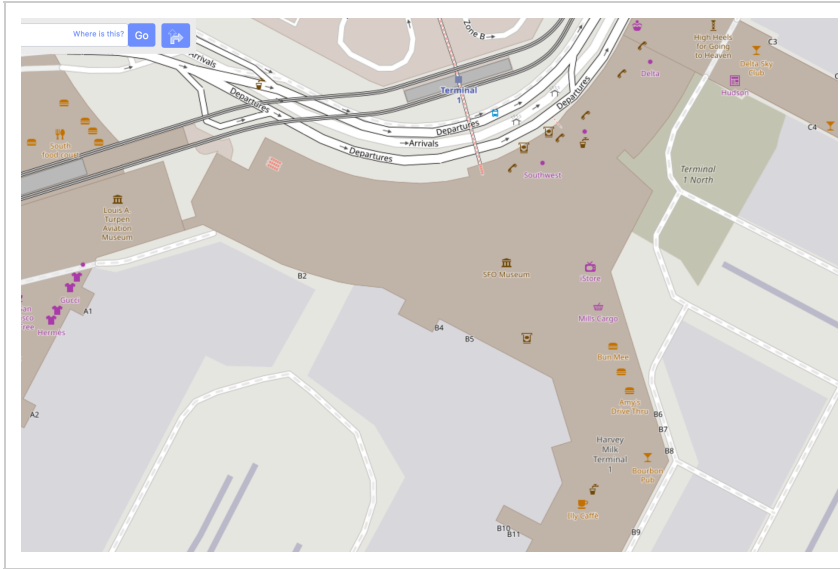


OpenStreetMap `https://www.openstreetmap.org/#map=16/37.6154/-122.3867` (OSM) has an up-to-date representation of SFO in 2022; however, we are not using OSM for our base maps, nor have we been able to justify the time of setting up a custom pipeline to process OSM data in to our own tiles. A cultural-heritage specific

set of OSM tools and services would be incredibly valuable and useful but, like everyone else, we're busy with other projects so it remains something to aspire to.

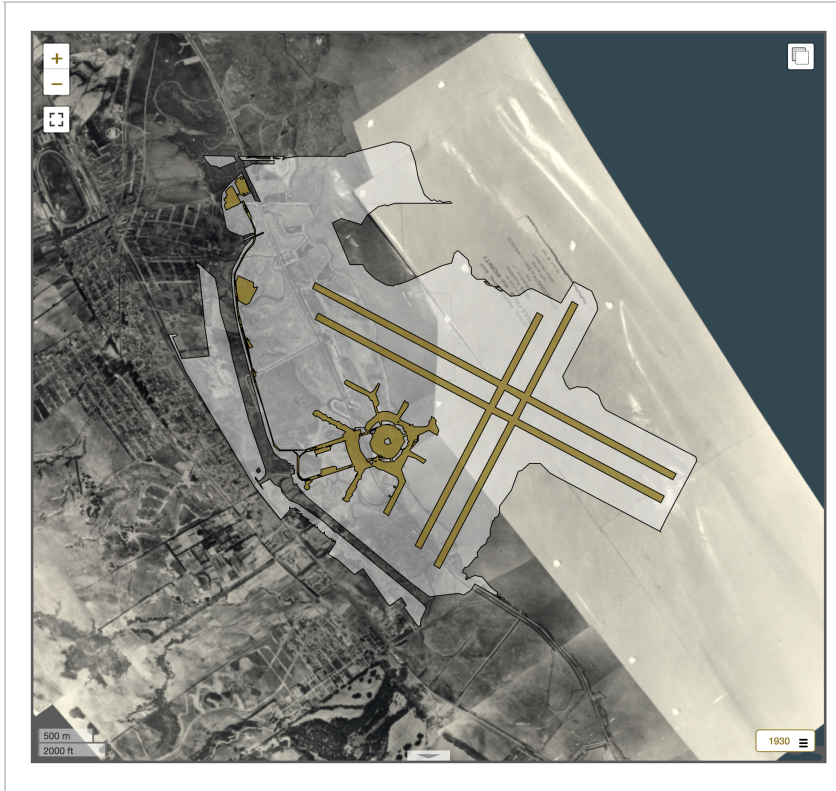


Even if we were to use the tiles that OSM publishes we would need a way to address the volume of interior data (shops, gates, inaccurate locations of SFO Museum installations, etc.) displayed on the map that we don't necessarily want to include in our maps. This is also true of both Apple's and Google's maps.



Our solution to all of these problems is straightforward: We simply hide what other people's base maps think about SFO by covering up that area with the footprint of the airport campus. Then we add a second layer made up of passenger-facing architectural elements, the AirTrain and the runways. As mentioned all of this data is produced by the airport itself so we can be confident that is accurate and up-to-date. This is very much a simplified representation of the airport but it is well-suited to the Museum's need and nothing prevents us from adding more detailed information in the future if we choose to.

All of these data about SFO are available as openly licensed data in the **sfomuseum-data/sfomuseum-data-architecture** `https://github.com/sfomuseum-data/sfomuseum-data-artchitecture` repository. These are the same data that we used in the **interactive map application in the T2 SkyTerrace** `https://millsfield.sfomuseum.org/blog/2021/11/19/skyterrace/` and we've adopted them for all our maps going forward. We've also updated all the maps to add the option to display historic aerial imagery if the map's view port is within the bounds of the SFO campus.

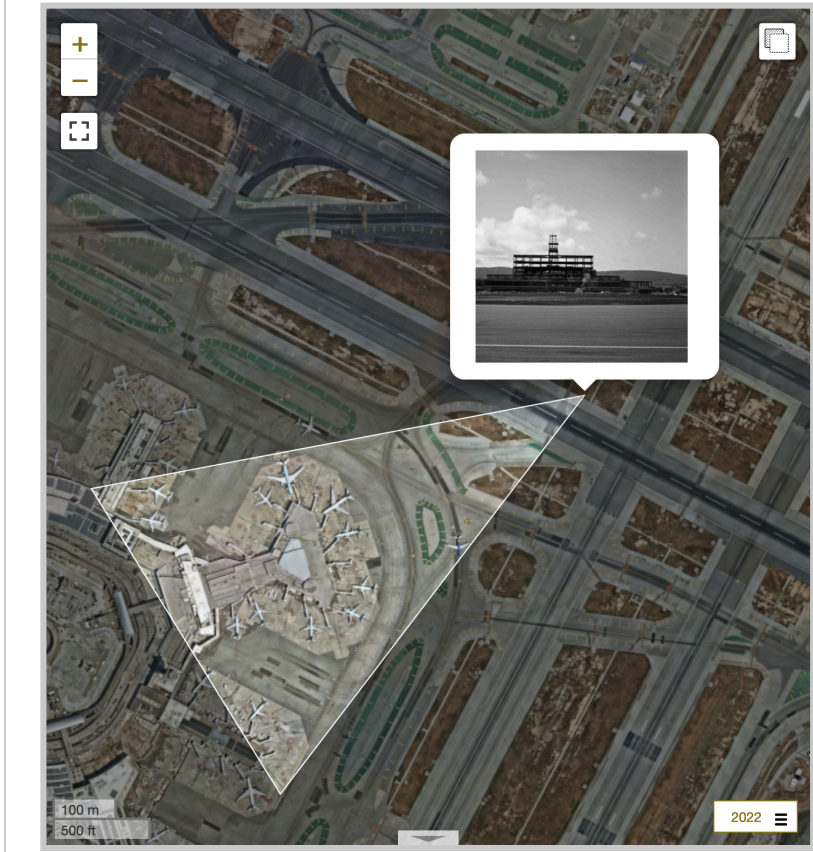


Here is a video of the flight path on arrival for **Southwest flight 2702** `https://millsfield.sfomuseum.org/flights/1780031137/` showing how historical aerial imagery is automatically loaded for a given map. It also highlights the new "resize" widget that is included with most maps allowing you to grow or shrink the height of the map.

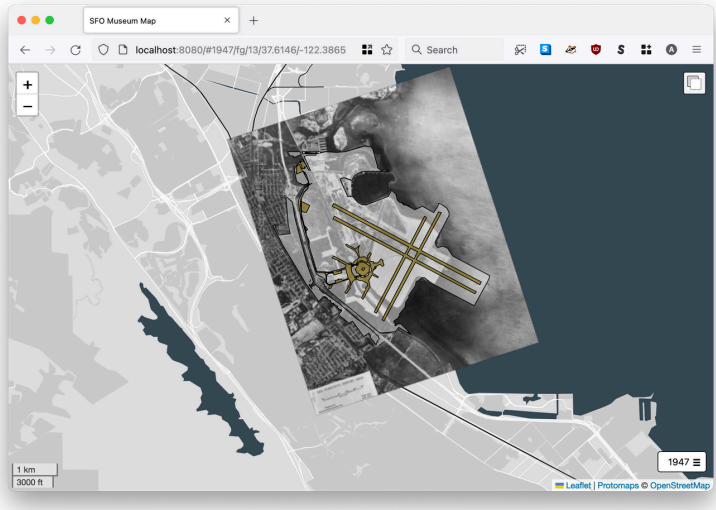
Did you notice the way the historical aerial layer defaulted to 2022? The updated map libraries will check to see if an HTML map element contains a data attribute containing a date and, if present, load the aerial layer closest to that date. For example the map

associated with this photograph of the **San Francisco Airport Terminal Building under construction** https://collection.sfomuseum.org/objects/1511943935/ in 1953 will automatically load the aerial imagery for **the airport in 1956** https://millsfield.sfomuseum/maps/1956 .

This is the same view **in 2022** https://millsfield.sfomuseum/maps/2022 .



We have created a dedicated project for all these new mapping libraries. It is called **www-sfomuseum-map** `https://github.com/sfomuseum/www-sfomuseum-map` and we use it for testing different base map providers, new features and improvements before rolling out updates to things like the **Mills Field** `https://millsfield.sfomuseum.org` or **Aviation Collection** `https://collection.sfomuseum.org` websites. This project is specific to the needs of SFO Museum but we are mentioning it here in the spirit of sharing how we do things.




```
$> go run -mod vendor cmd/server/main.go \
        -map-provider protomaps \
        -nextzen-apikey {PROTOMAPS_APIKEY}
```

"Cake Plane"

On a personal note I am especially excited about the 2022 aerial imagery because it appears to have captured **the aircraft I call "Cake Plane"** `https://millsfield.sfomuseum.org/map#2022/fg/20/37.63562/-122.38719` visiting SFO. "Cake Plane" is a 747 cargo plane, with the tail number `N405KZ`, operated by Air Atlas. Notably it is **painted white from nose to tail** `https://flightaware.com/photos/aircraft/N405KZ` which makes it stand out at an airport full of passenger airlines and their colorful liveries.

One day, a few years ago, I was **waiting for the**

AirTrain `https://millsfield.sfomuseum.org/id/1763588577/` and noticed this plane parked nearby. At the same time I caught the scent of chocolate cake wafting powerfully through the air. It stands to reason that the scent of chocolate cake was coming from the nearby kitchens preparing food for all the passenger flights departing SFO. However, I chose to imagine that all that cake was actually being baked inside the belly of this mysterious white jumbo jet; that if you opened up the nose of the plane you would be presented with a 747-sized piece of cake!

It's not possible to see the tail number of the white 747 captured in 2022 so I am not certain whether this aircraft is actually "Cake Plane" but I like to think that it is.

2022-08-08

can you have a social contract without a passing lane?

Redwood #4 (2022)

Redwood #4 (2022)

23.5 by 68 inches. Chiyogami and paper and acrylic on fabric. I am still trying to decide how to hang this.

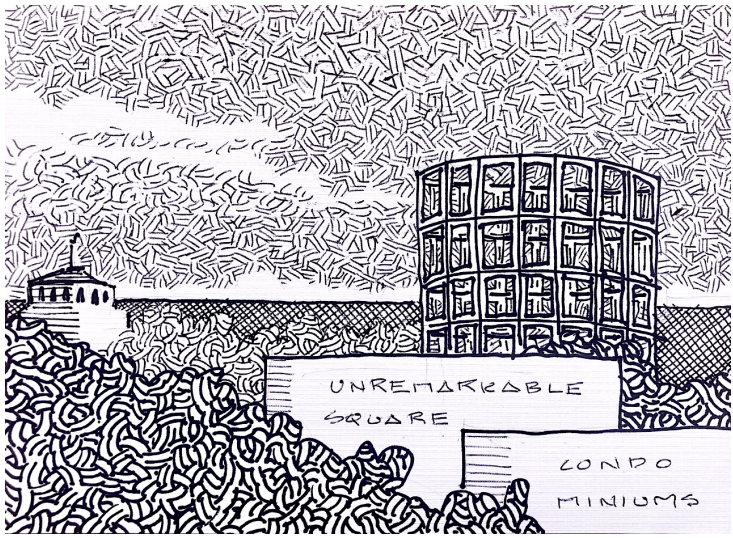
2022-09-23

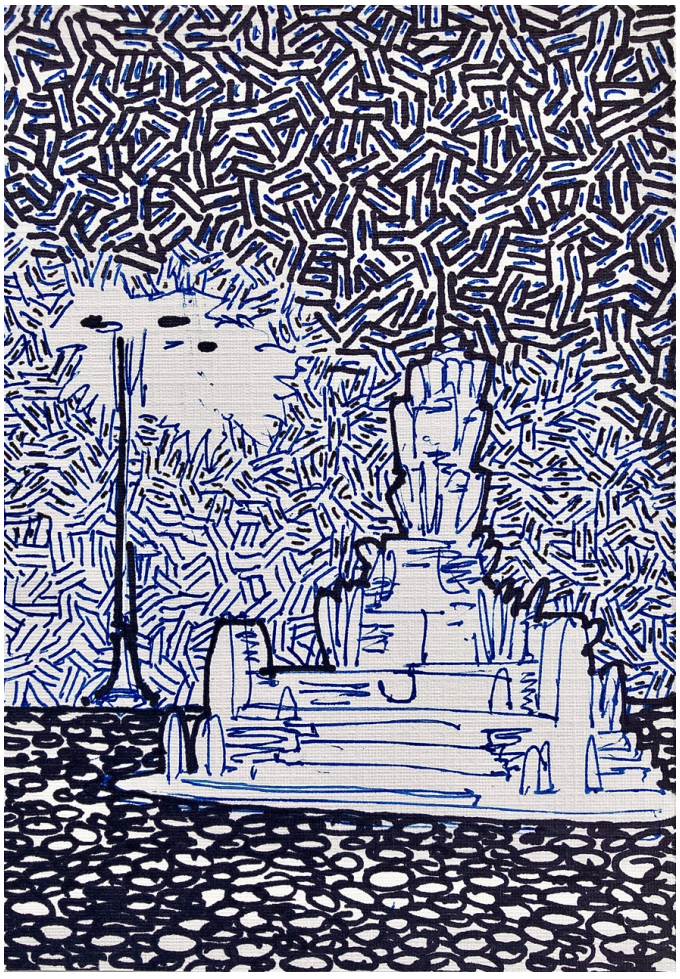
the personal trainers of villa borghese

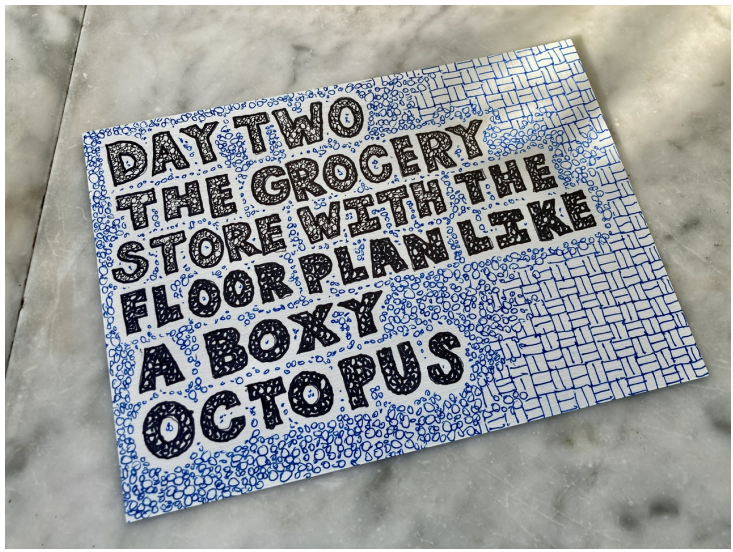
Postcards of Italy

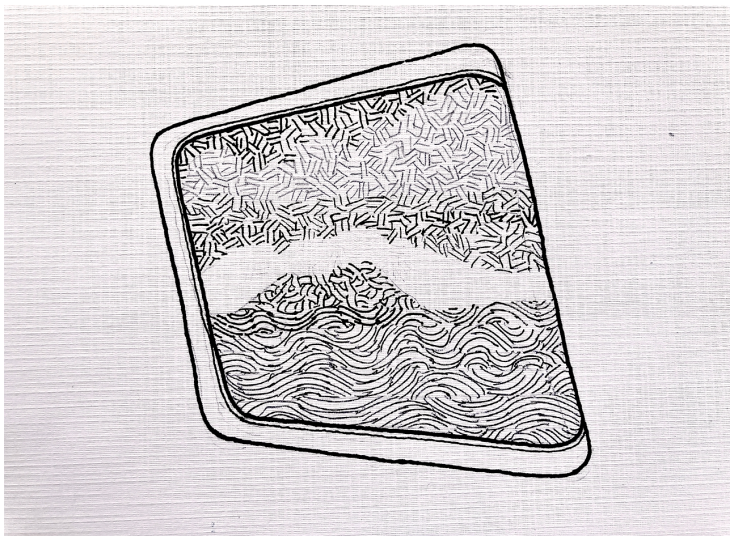
Postcards of Italy

We went to Italy recently. These are some of the postcards I made and sent to people along the way.


UNREMARKABLE
SQUARE
CONDO
MINIUMS

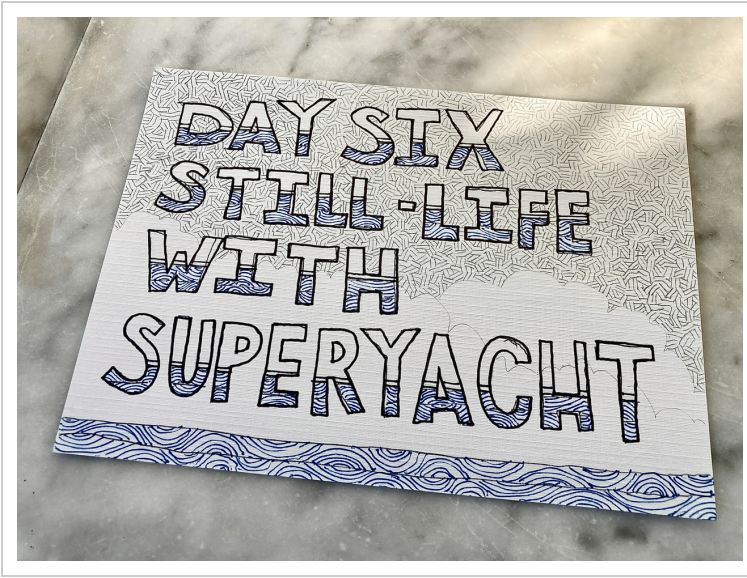



DAY TWO
THE GROCERY
STORE WITH THE
FLOOR PLAN LIKE
A BOXY
OCTOPUS

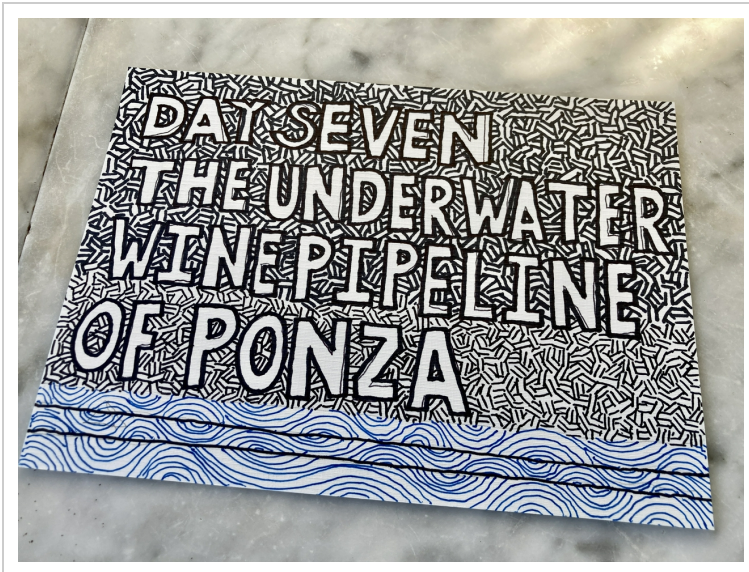


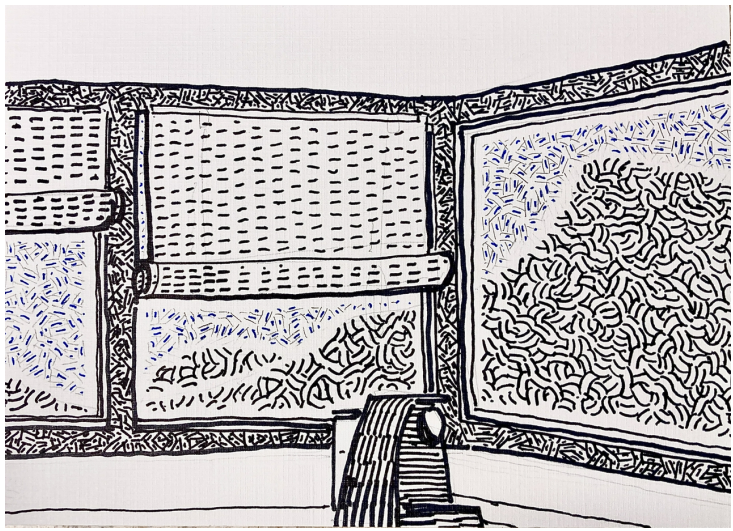
Mister Big

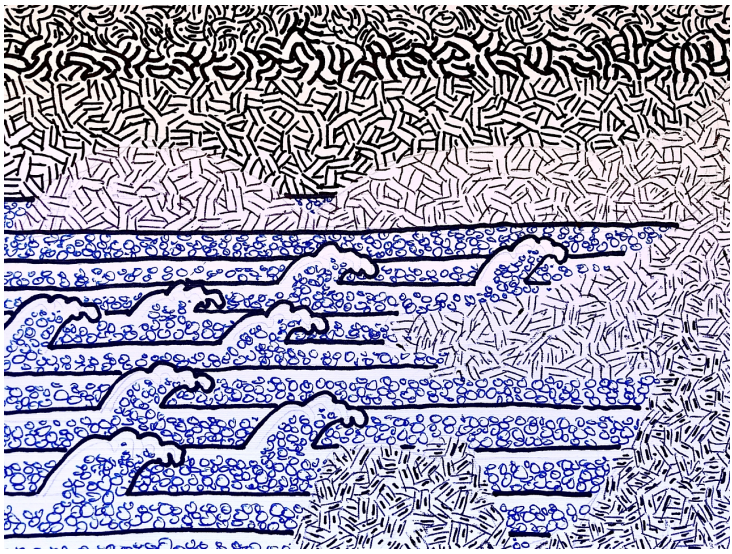
Balls `https://www.aaronland.info/weblog/2019/03/19/signs/#mrbigballs` is everywhere.



There really was a superyacht that came and parked itself in the same cove that our house overlooked. Eventually we learned it was the **Maltese Falcon** `https://en.wikipedia.org/wiki/Maltese_Falcon_(yacht)` which you can rent at a starting rate of 460,000€ / week. We did not have binoculars so it was hard to see what was going on but the absence of any movement on deck suggested they weren't having much fun. Maybe they were just there to complete some nefarious plan like in a James Bond movie.


DAY SEVEN
THE UNDERWATER
WINE PIPELINE
OF PONZA

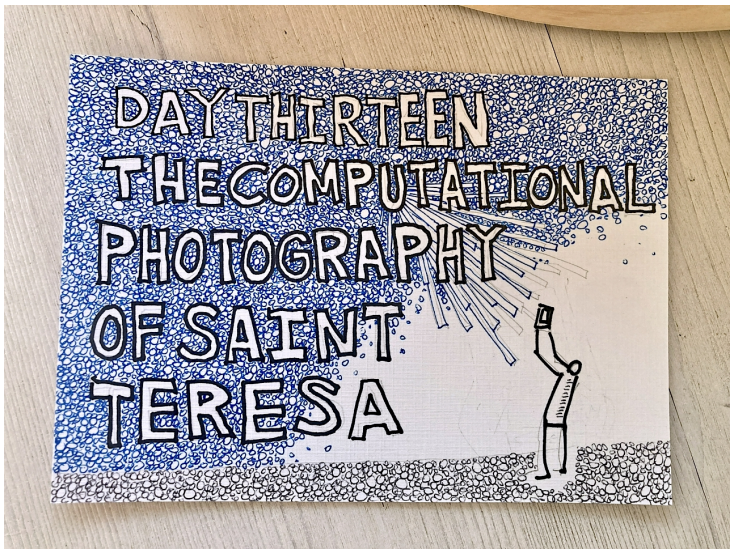


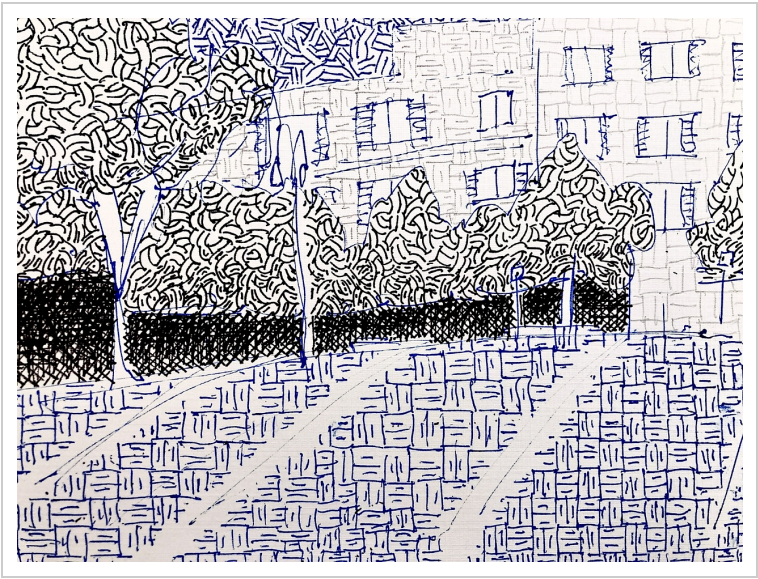


DAY ELEVEN
ART STUDENTS
DRAWING
EVERYTHING BUT
THE TOURISTS

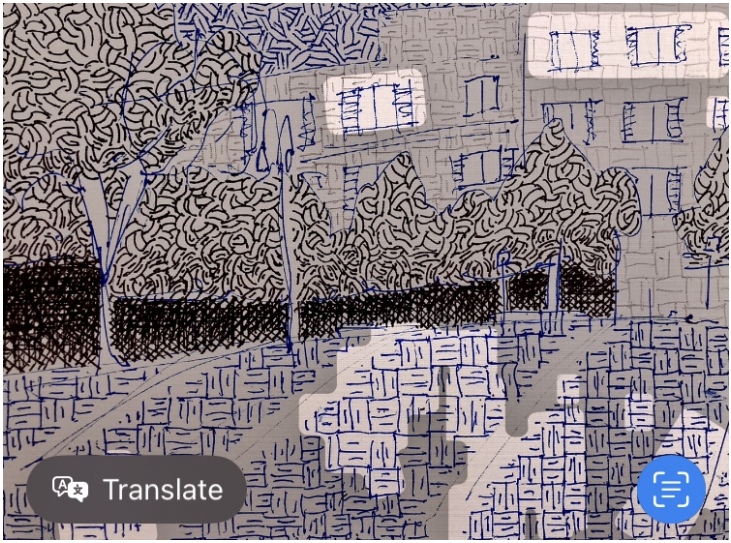
DAY TWELVE
APPROACHING
PEAKBREAKFAST
COOKIE

WAZZZZUPPP!!!

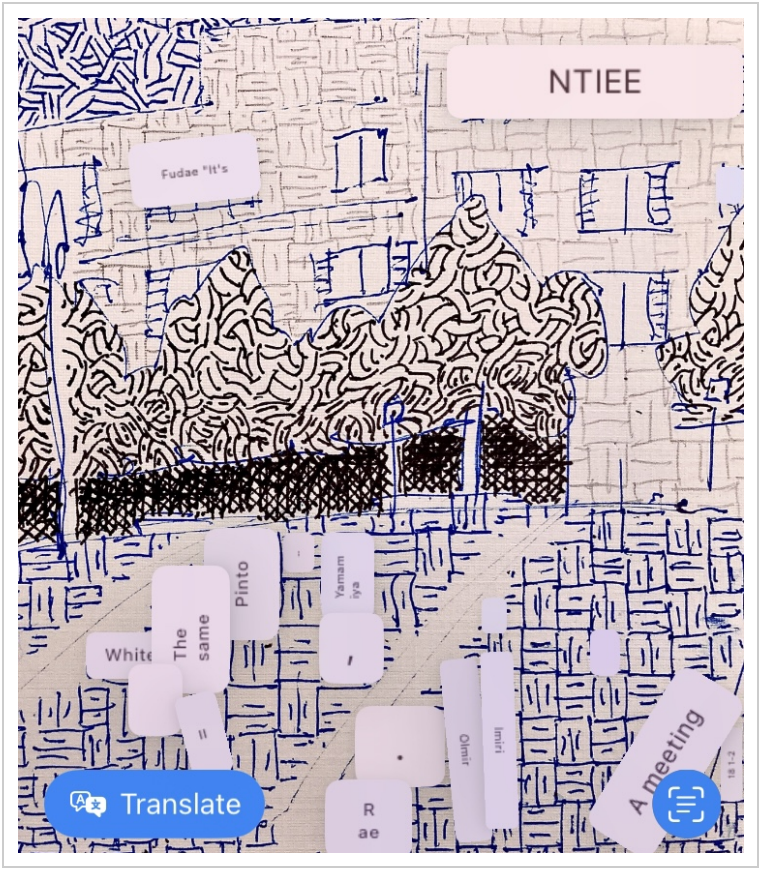

DAY THIRTEEN
THE COMPUTATIONAL
PHOTOGRAPHY
OF SAINT
TERESA



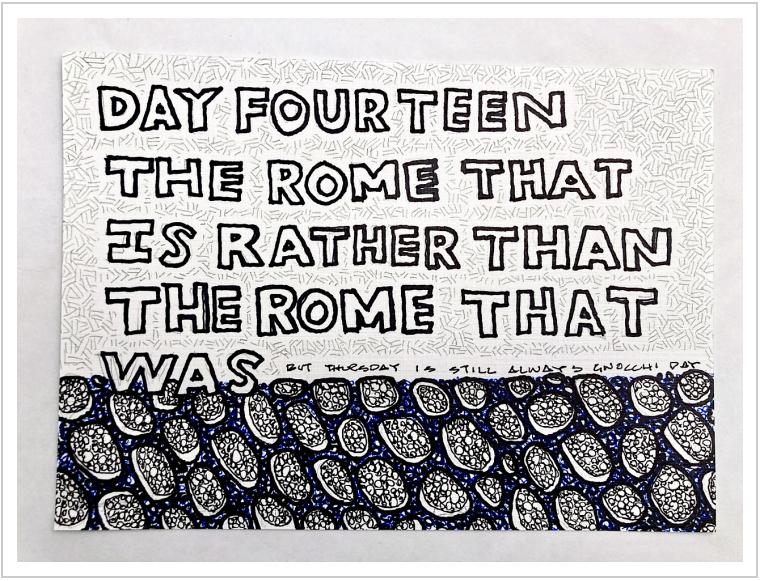
The computer vision stuff on my phone is a little obsessed with these drawings.



It seems to "think" or wants to "believe" that there's a whole other meaning in them.



For example... this.



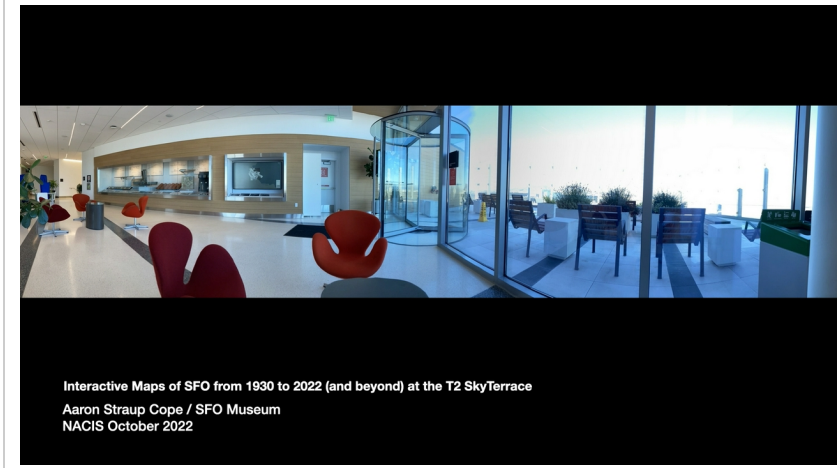
Have you ever noticed the way the cobblestones in Rome look like they could be sculptures of large pieces of gnocchi? Coincidence?

2022-10-10

things I have written about elsewhere #20221028

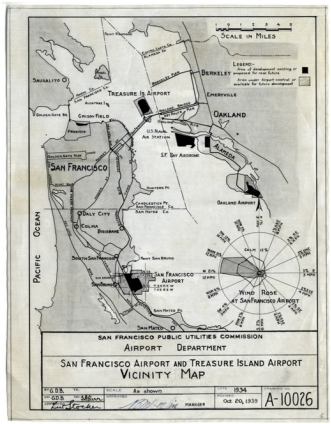
Presenting the SkyTerrace Interactive Map at...

Presenting the SkyTerrace Interactive Map at NACIS 2022



This was **originally published on the Mills Field weblog** `https://millsfield.sfomuseum.org/blog/2022/10/28/nacis/` , in November, 2022.

These are the slides and notes for the presentation I did at the 2022 **North American Cartographic Information Society** `https://nacis.org` (NACIS) conference in Minneapolis. The talk, titled "Interactive Maps of SFO from 1930 to 2022 (and beyond) at T2 SkyTerrace", discusses the technical requirements, constraints and decisions made building **the interactive map application on display in the SkyTerrace Observation Deck** `https://millsfield.sfomuseum.org/blog/2021/11/19/skyterrace/` in Terminal 2.



www.sfomuseum.org

collection.sfomuseum.org

millsfield.sfomuseum.org

@thisisaaronland

Hi, my name is Aaron. I work at the **San Francisco International Airport Museum** `https://sfomuseum.org` . The airport is owned and operated by the City and County of San Francisco. In 1980 it created an arts program which eventually became the museum. At the moment there are 25 active galleries located throughout the terminals and since its founding the museum has produced over **1,800 exhibitions** `https://millsfield.sfomuseum.org/exhibitions/` . The museum also has a **permanent collection of 150,000 objects** `https://collection.sfomuseum.org` related to the history of the airport and commercial aviation. **We have an amazing collection of airsickness bags** `https://collection.sfomuseum.org/objects/subcategories/1511213185/with-images/` .

If you didn't know this, or were unaware of the scope of the museum's work, don't worry. Most people don't and part of my job,

as I see it, is to use the internet (and a lot of maps) to help fix this problem.

I started writing my notes for this talk, in long form, last week and noticed that I had written 2,500 words before reaching the halfway point. So, this is not going to be that talk. I am going to skip from peak to peak dipping in to the details only when it seems necessary. Which means there may be places during this talk where you are left wondering "But, what about..." If you do I encourage you to come find me after the talk and I will be happy to answer your questions.

In 2018 we made a web application, hosted on the museum's Mills Field website, **of old aerial imagery of SFO** `https://millsfield.sfomuseum.org/map` . It's a pretty standard **Leaflet-based web application** `https://leafletjs.com/` with imagery from as early as 1930 all the way up to 2022 sourced from the museum's collection, the airport's GIS department and the special collections library at UC Santa Barbara.

Fun fact: SFO has its own map projection. (It has two, actually.)

Our short-term goal is to eventually have imagery for every year since 1927 in order to explain how the airport has grown and changed. Our long-term goal is to do this for every other airport that SFO holds hands with.

Another fun fact: There is only **one structure from 1956** `https://millsfield.sfomuseum.org/terminals/1159396131/` still standing at SFO: **The arrivals and departures concourse at Terminal 2** `https://millsfield.sfomuseum.org/commonareas/1763588165`

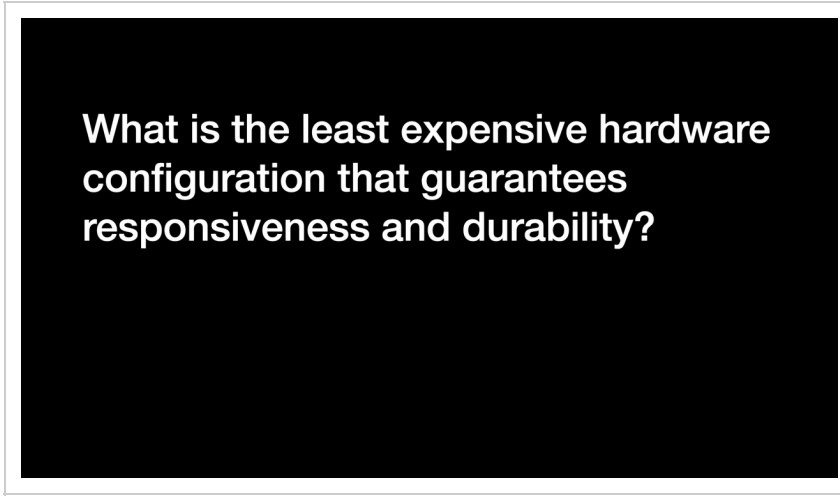
/ which is, coincidentally, where the SkyTerrace observation deck is located. At the end of 2018 I was asked whether we could to turn that web application in to a touch-based interactive installation to be included in that same SkyTerrace Observation Deck set to open in February of 2020.

This is a photo of some early work done to imagine what the interface for that application would look like, when we were thinking that we might use a single, large touch-enabled monitor mounted to the wall. Had we done that however Americans With Disabilities Act (ADA) requirements would have meant that only the bottom third of the display was available for use as an interactive surface. The remaining two-thirds of the screen, including the arrows in this example, would have been out of reach of people in wheelchairs.

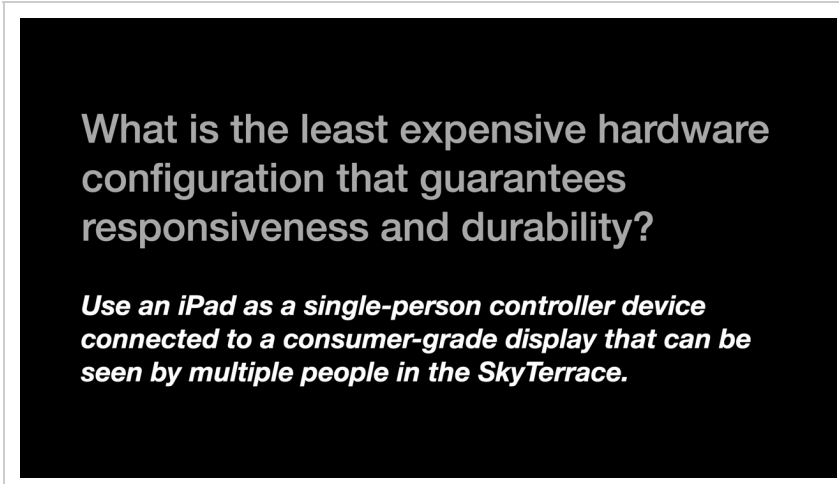
This image shows the **SFO of 2019** `https://millsfield.sfomuseum.org/map/2019` overlayed on top the **SFO of 1946** `https://millsfield.sfomuseum.org/map/1946` . Do you see that angled terminal at the bottom of the screen? That was demolished shortly after this photo was taken to finish construction on the new Harvey Milk Terminal 1 building and is a good illustration of why we are so keen to document, and map, the history of the airport.

Two more fun facts: One, SFO Museum maintains an openly licensed **catalog of historical architectural footprints** https://github.com/sfomuseum-data/sfomuseum-data-architecture of the airport but we don't have time to talk about that today. Two, the old non-existent Terminal 1 building is **still on Google Maps in 2022** https://millsfield.sfomuseum.org/blog/2022/08/08/maps/ .

One thing worth mentioning is that the airport already had a history of hiring outside firms to design and develop these kinds of digital, interactive installations in the past. Those experiences haven't always end well so when I started thinking about how to transition the work we'd done online to the physical world I was keenly aware that people were wary whether any of this would work. With that in mind the first part of the design process for developing an interactive map was to answer four questions:



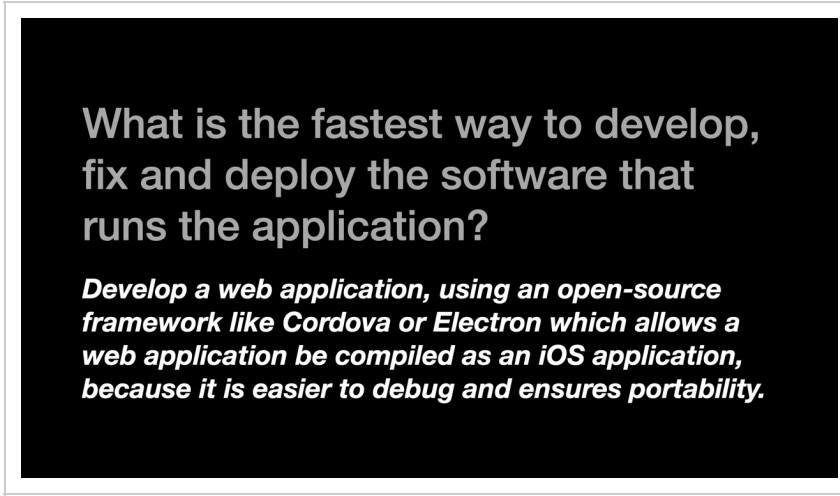
What is the least expensive hardware configuration that guarantees responsiveness and durability ?



Use an iPad as a single-person controller device connected to a consumer-grade display that can be seen by multiple people in the SkyTerrace.

On the face of it iPads are not the cheapest devices on the market today. But when you factor in the quality and responsiveness of touch support in iOS, the durability of the screens, the processing power and available storage and the overall stability of long-running applications and then compare the iPad to other consumer and other vendor offerings the price no longer seems so bad. Importantly it is a fixed price and there are reasonable assurances that the same amount of money will buy better hardware, or that less money will buy equivalent hardware, in the future.

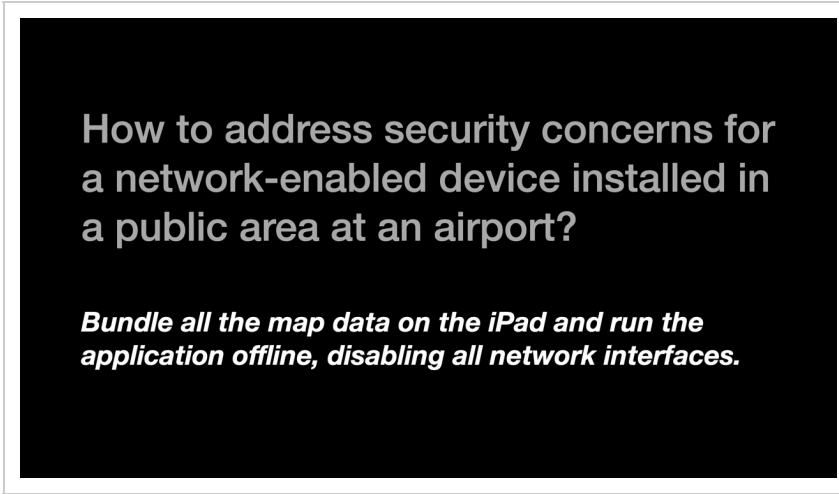
What is the fastest way to develop, fix and deploy the software that runs the application?



Develop a web application, using an open-source tool like **Cordova** `https://cordova.apache.org/` or **Electron** `https://www.electronjs.org/` that can be compiled as an iOS application, because it is easy to debug and ensures portability.

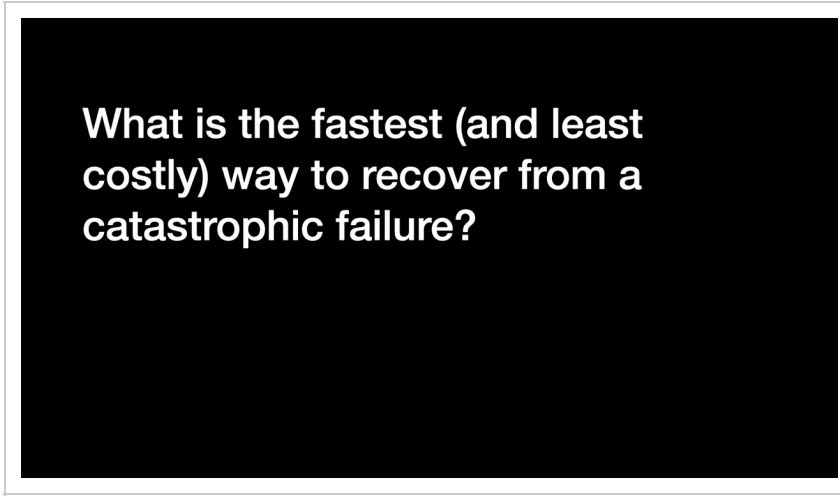
Developing native applications for either of the two major platforms, iOS and Android, has often been described as death by a thousand cuts. It is a time-consuming process and often even minor updates are time-consuming and labourious. On the other hand SFO Museum already had an existing web application so why not re-use that? Even though there would inevitably be changes to accomodate an interactive kiosk the core of the application would remain the same. Importantly developing, and debugging, web applications is easier and faster than native applications. Wrapping that application in a framework like Cordova or Electron would allow us to target not just iOS but Android devices as well which would be an important buffer against vendor lock-in.

How to address security concerns about a network-enabled device installed in a public area at an airport?

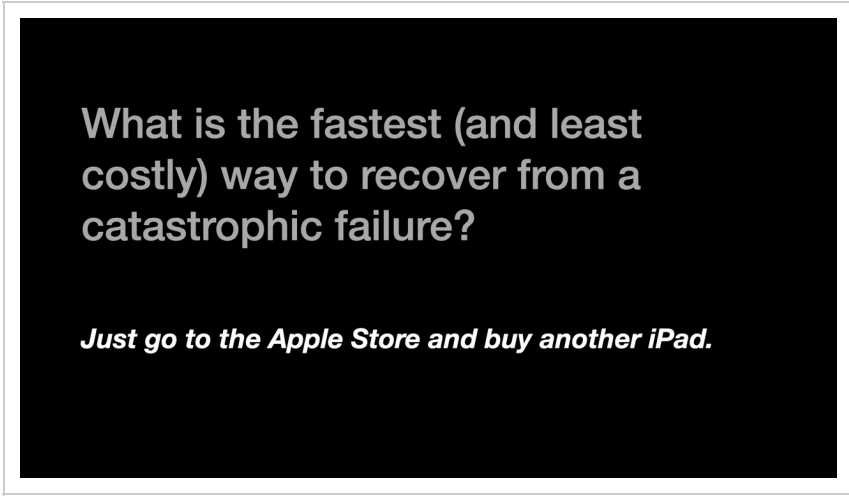


Bundle all the map data on the iPad and run the application offline, disabling all network interfaces.

The decision to make the application work offline was a pragmatic one. Any device and especially a network-enabled device deployed on the airport campus, particularly in public areas, is subject to rigorous scrutiny. That scrutiny, though warranted, demands time and resources from other departments which means everything takes longer to do because it is work that is scheduled and prioritized relative to all the other things happening at a busy airport. Rather than fetching all the map tiles necessary to display the historic aerial imagery over the network we simply bundled those tiles on, and read them from, the iPad itself.



What is the fastest (and least costly) way to recover from a catastrophic failure?

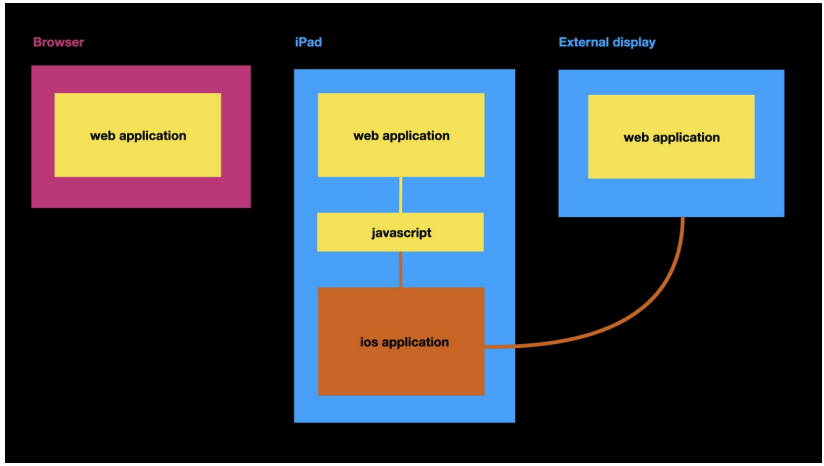


Just go to the Apple Store and buy another iPad.

I mentioned that the airport and the museum are owned and operated by the City and County of San Francisco. As such we can't "just go to the Apple Store and buy another iPad" because we have rules and processes for spending that kind of money. But it points to the idea of ensuring that the "**time to recovery** `https://www.kitchensoap.com/2010/11/07/mttr-mtbf-for-most-types-of-f/` " when a catastrophic failure occurs (because it will) is as short as possible. If necessary it would be easier and faster to find a way to authorize a one-time purchase of a consumer device, available at a variety of outlets within driving range of the airport, than to deal with creating a purchase order for a vendor and then waiting for it to be fulfilled.

So that's what we did. And everything seemed to be going according to plan until we realized that we couldn't use either Electron or Cordova to package our web application. The earliest versions of the application had been simply been mirroring the iPad to the external display but it was decided that we did not want to show the interface elements used to control the map on the external display.

At the time, I was surprised to discover that the only way to show different content on two screens from a single application was to use native iOS frameworks. This might still be the case. At this point it felt like there was a real risk that we would get dragged in to the quicksand of native application development.

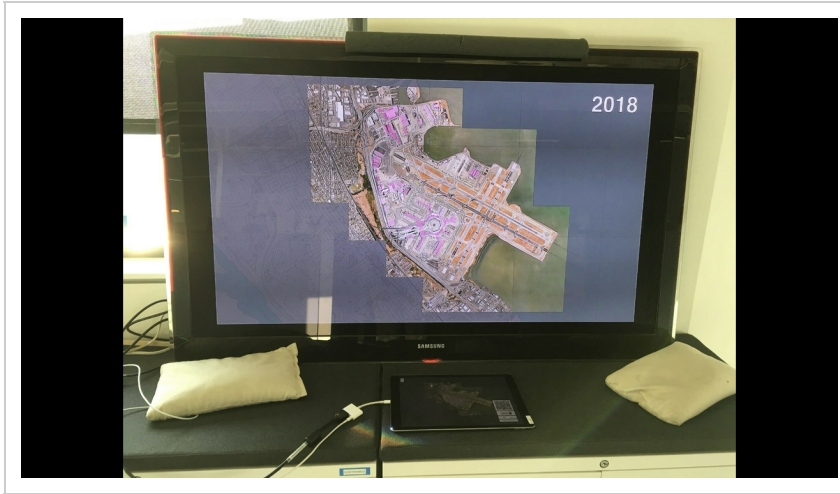


Fortunately native mobile application frameworks make it possible **to embed and use a web application as a first class interface element** `https://developer.apple.com/documentation/webkit/wkwebview` . Importantly, in iOS, it is also possible to **relay messages** `https://developer.apple.com/videos/play/wwdc2020/10188/` between the web application and the native parent application using JavaScript.

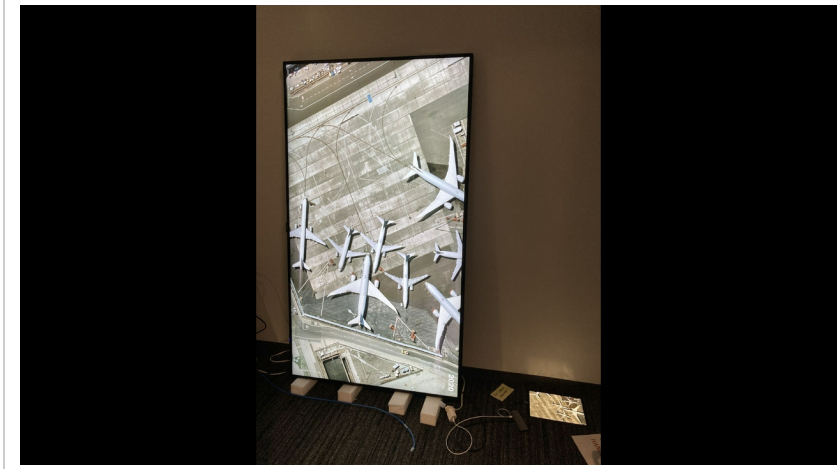
What this meant is that we could continue to use the original web application with some minor changes to send Leaflet map events from the web application back up to its parent iOS application which would then transit them down to the equivalent web application running on the external display.

We made a bucket brigade to relay and synchronize Leaflet map events, rather than touch events, between two different web applications. That's kind of the whole trick but it's a good one and

when you realize that it's really an interface for exposing native iOS functionality to embedded web applications it opens up a world of possibilities.



And it worked. Importantly, we had also created a generic framework for developing web applications that could, with predictable and manageable costs, run offline and be deployed to any 4x4 foot space at the airport. When I talk about costs I'm not talking about the five or six figure budgets that external agencies might charge for something like this. I'm talking low four figures plus the cost of application development and building furniture to house the installation.

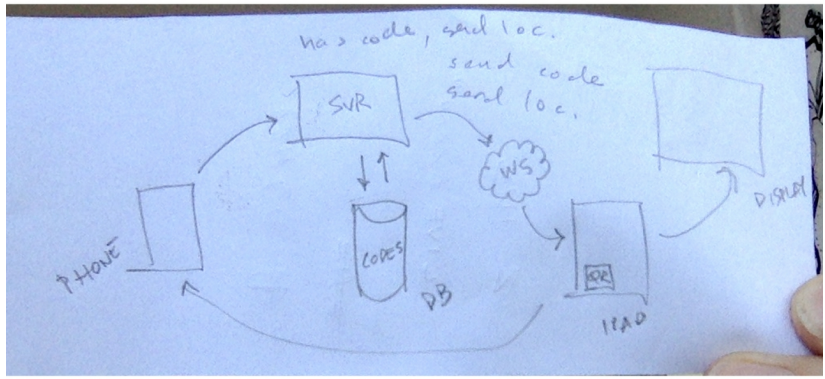


I mentioned that the SkyTerrace was set to open in February 2020 but that was eventually pushed back to later in the year. And then COVID-19 happened. Passenger traffic cratered, everyone started working from home and, not surprisingly, projects like the SkyTerrace were put on hold. The satellites were working, though, so SFO still got aerial imagery that year and **this is what the airport looked in July, 2020** `https://millsfield.sfomuseum.org/map#2020/fg/17/37.63310/-122.39290` .

The good news is that, when the airport and the museum began to resume on-site operations in earnest, the application I'd developed had been running unattended and attached to a big honking monitor for 18 months and everything still worked. Go me!

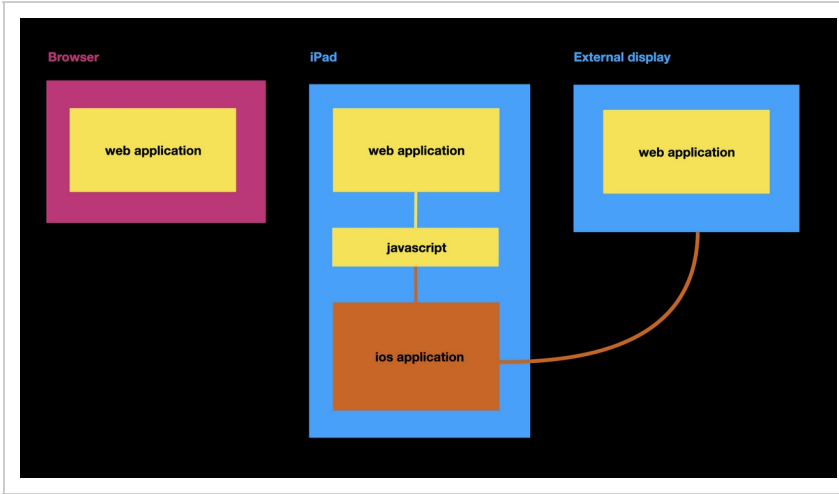
The bad news was that no one wanted to touch any kind of public surface anymore. Whether or not that concern was justified was

irrelevant so far as the airport was concerned. We meet the passengers where they are, so to speak, and I was asked: Could the installation be controlled by an individual passenger's phone or mobile device instead of their finger? Oh, and also, the SkyTerrace was going to open in a month or so.

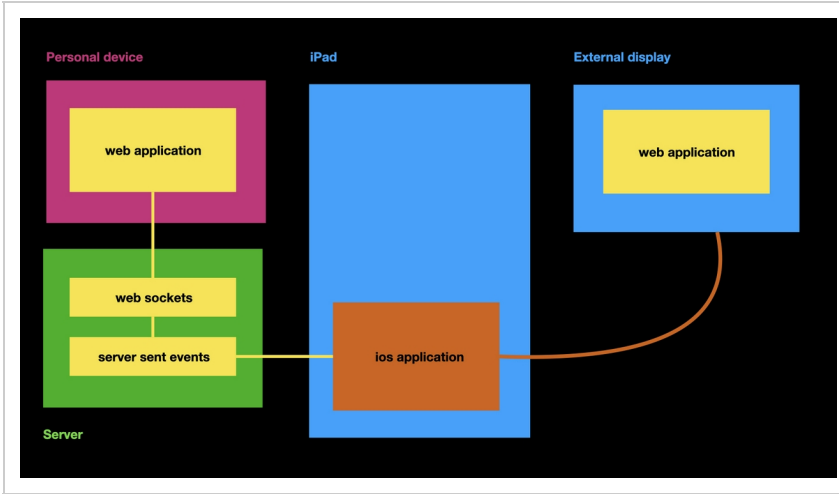


My first concern was figuring out how to rewrite as little of the original application as possible. There simply wasn't time for anything else. Do you remember the way I mentioned that the original touch-based application was relaying Leaflet events to iOS using the JavaScript bridge? It turns out the crux of the problem of, as well as the solution to, transitioning things to an application controlled by a person's phone was simply replacing that bridge with a network-based equivalent.

This is the original design sketch I made when I started thinking about the problem.



As a reminder, this was the original design for how messages were brokered between the iPad and the external display.



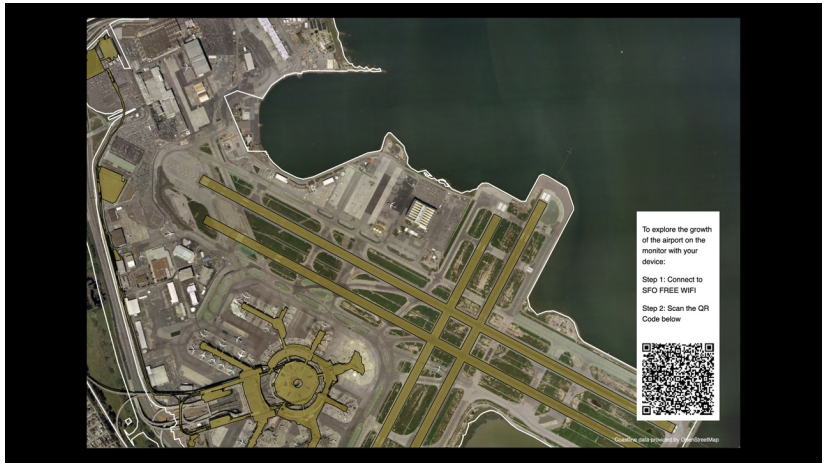
This was the updated design. In order to accomplish this we would need to run a web server that performed two functions.

On the one hand it would serve the same map application we'd already developed, but to a passenger's phone. It would capture the Leaflet map events as usual and send them back to the server using **WebSockets** #, a web standard for bi-directional communication between parties (in this case the server and a person's browser).

On the other hand the server would parrot those same events back to the iPad using **ServerSent Events** #, another web standard for unidirectional communications, which would then pass the messages to the application running on the external display as it always had.

This was the same bucket brigade sending events to the application running on the external display but with a few more participants. Likewise none of the hardware changed. We just hid the iPad but it

was still there connected to the external display and doing all the heavy lifting.



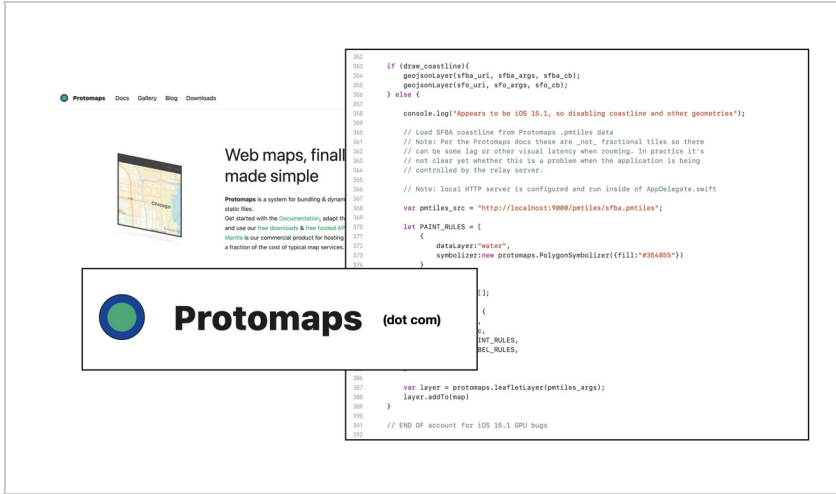
Control of the map is goverened by passengers scanning a QR code which is displayed on the external display. This contains a URL and time-sensitive access token which, when scanned, causes the QR code to disappear, and grants control of the map to that passenger for five minutes. After five minutes a new QR code is displayed which allows another passenger to assume control but the old access token will continue to work until then.

All of which introduced a host of new software, hardware, network and security dependencies but in September 2021 the SkyTerrace observation deck opened to the public along with a working hands-free interactive map application.

And, amazingly, everything worked fine. At least until iOS 15 was released.

You may not remember but it was eventually determined that, for security reasons, iOS 14 is basically unsafe at any speed. I was not going to be the guy who left unpatched hardware with known security issues on the floor in the airport so I diligently updated the iPad running the interactive installation. Whereupon the application was no longer able to render the San Francisco Bay Area coastline, an important graphic element we use to contextualize the map.

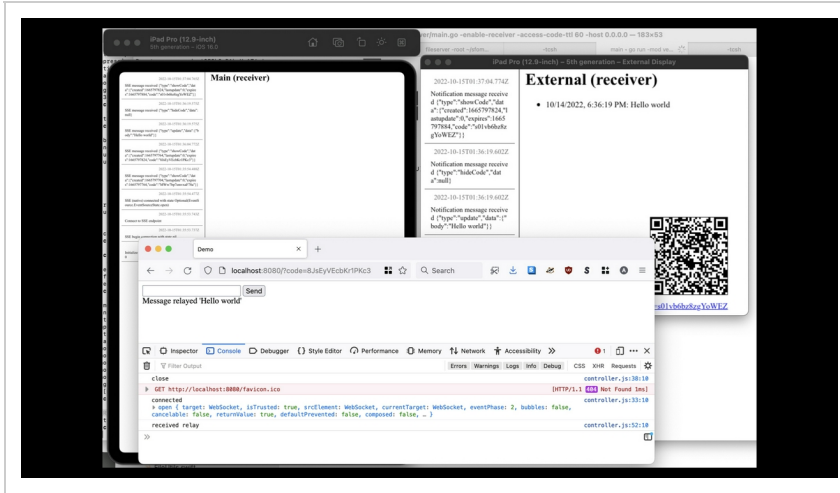
Worst of all, we actually knew about this problem before we updated the iPad. The specific problem is iOS 15's inability, in web applications, to render geometries with more than a certain number of vertices. We were aware of the issue throughout the beta releases of iOS 15 but hoped it would be remedied with the final release. It was not and would not be fixed until iOS 16. I mention this to highlight what I still see as the constant high-wire act that developing native applications for mobile platforms continues to be.



In the end we were able to work around the iOS 15 problem using Brandon Liu's **Protomaps** `https://protomaps.com` . I don't have time to get in to that here so I will just say that I think Protomaps is the most exciting thing to happen in digital mapping in the last ten years.

Throughout these changes I have made sure to keep the old application in place such that the installation can be operated using a touch-based or a touch-less model with the flip of a switch. I recognize that people, present company included, are still wary of touching public surfaces but I also understand that those concerns are probably unwarranted given what we know after living with COVID-19 for nearly three years.

My own feeling is that an application you can control from your phone, without having to install any additional software, is kind of cool both as a user and a developer but that the original touch-based interface was both more inviting and more rewarding. It is unclear whether we will ever go back, though, so it's good to know we can develop these kinds of applications either way.



Finally, I like to believe there is always a larger meta-project in any given endeavour.

For me that has meant doing the extra work to share not the specific implementation of any one SFO Museum initiative but to decouple the underlying tools and software components that might be repurposed in other like-projects. The cultural heritage sector, and more broadly anything that might be dubbed the public sector, does not enjoy the capital resources or the head count of the private sector and, at least for the foreseeable future, never will.

No one else is going to save us and one of things we need, in order to meet the public's expectations in a world of digital and networked technologies, is a better kit of tools. Not the big, everything-but-the-kitchen-sink initiatives of the past but the all the different kinds of hammers, screwdrivers, chisels and other tools that we might use to fashion specific solutions tailored to our individual needs.

In November, 2020 **we released a multiscreen "starter kit" package** `https://millsfield.sfomuseum.org/blog/2020/11/18/multiscreen/` to demonstrate the basic bucket brigade message-relaying iOS application I've been describing. In a week or two we will release an updated version of that package demonstrating how to set up a network-enabled bucket brigade, along with an equivalent starter kit for the server component, so that you might develop your own "touch-free" applications.

https://github.com/sfomuseum
https://github.com/sfomuseum-data
@thisisaaronland

Thank you.

As mentioned at the end of the talk we have published open source reference implementations (or "starter kits") for both the updated **iOS application** `https://github.com/sfomuseum/ios-multiscreen-starter/tree/main#hands-free` and the **relay server** `https://github.com/sfomuseum/www-multiscreen-starter` . These applications are *not* the map application I presented at NACIS. They are deliberately simple and minimalist example applications to demonstrate how to dispatch messages between a variety of sources: A mobile device, a server, an iOS application and finally multiple "views" in that application. What messages you send and what your application does with them will be specific to your needs.

1. ***https://github.com/sfomuseum/ios-multiscreen-starter/tree/main#hands-free*** `https://github.com/sfomuseum/ios-multiscreen-starter/tree/main#hands-free`

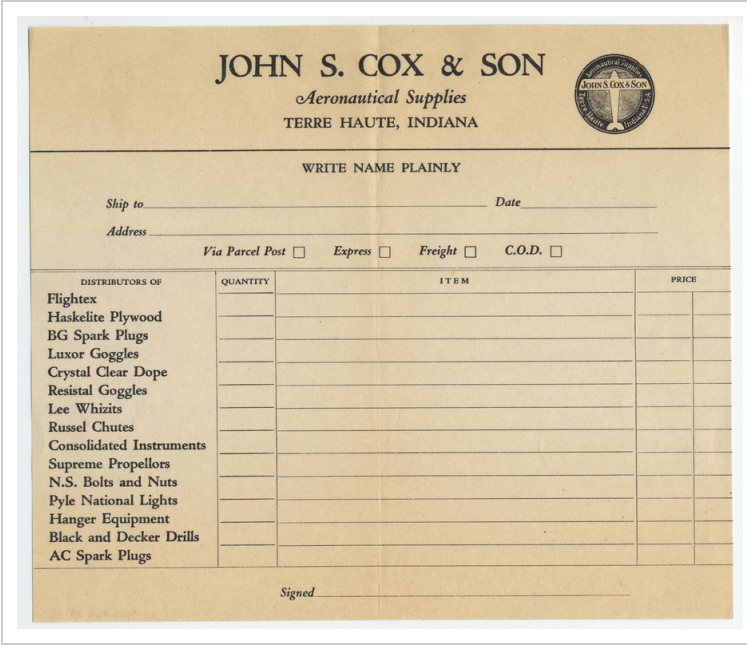
2. ***https://github.com/sfomuseum/www-multiscreen-starter*** `https://github.com/sfomuseum/www-multiscreen-starter`

things I have written about elsewhere #20221114

Updating the Who's On First Browser to support...

Updating the Who's On First Browser to support Tailscale and Protomaps

This was **originally published on the Who's On First weblog** `https://whosonfirst.org/blog/2022/11/14/browser/` , in November, 2022.



JOHN S. COX & SON

Aeronautical Supplies

TERRE HAUTE, INDIANA

WRITE NAME PLAINLY

Ship to _____ *Date* _____

Address _____

Via Parcel Post ☐ *Express* ☐ *Freight* ☐ *C.O.D.* ☐

DISTRIBUTORS OF	QUANTITY	ITEM	PRICE
Flightex			
Haskelite Plywood			
BG Spark Plugs			
Luxor Goggles			
Crystal Clear Dope			
Resistal Goggles			
Lee Whizits			
Russel Chutes			
Consolidated Instruments			
Supreme Propellors			
N.S. Bolts and Nuts			
Pyle National Lights			
Hanger Equipment			
Black and Decker Drills			
AC Spark Plugs			

Signed _____

Catalog: John S. Cox & Son, aeronautical supplies c. 1934. Purchase. **Collection of SFO Museum**. 2002.064.041 a e

The **Who's On First Browser** `https://github.com/whosonfirst/go-whosonfirst-browser` project was a by-product of the **Great Mapzen Shutdown** `https://www.whosonfirst.org/blog/2018/01/02/chapter-two/` of 2018. At the time there was some question about whether we'd be able to keep the **Who's On First Spelunker** `https://spelunker.whosonfirst.org` running

without the support of Mapzen.

In the end everything worked out. The Spelunker is still running, today, but the experience highlighted the importance of having a ready alternative on hand. Something inexpensive and easy-to-maintain which, absent a searchable index, made sure there were still **human, machine readable and graphical representations** `https://whosonfirst.org/blog/2019/12/20/browser/` for every Who's On First ID, with links to their relations, available on the web. That tool was the Who's On First Browser.

This post is about some recent, optional, features we've added to that tool: The ability to run it as a **Tailscale virtual private service** `https://tailscale.com/blog/tsnet-virtual-private-services/` and to use **Protomaps** `https://protomaps.com` for display maps.

A quick recap

The basic requirements for the Who's On First Browser were:

1. ***It would be written in Go.***

2. ***It could be run as a standalone web server or as an AWS Lambda function attached to an AWS API Gateway instance.***

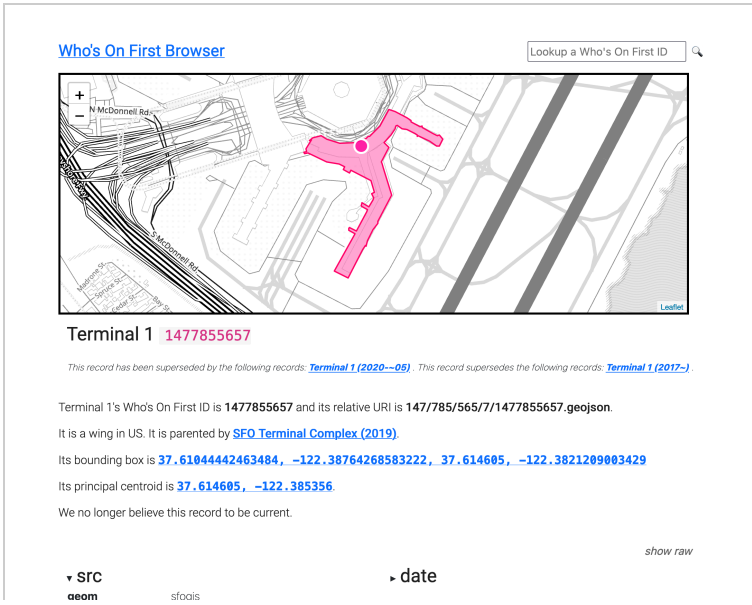
3. ***It would read and render individual records from any valid "reader" source, typically the local file system or something like an AWS S3 bucket.***

For example, something that could be started as easily as:

```
$> bin/whosonfirst-browser \
        -enable-all \
        -reader-uri repo:///usr/local/data/sfomuseum-data-architecture \
        -nextzen-api-key {NEXTZEN_APIKEY}
```

The `sfomuseum-data-architecture` *repository is not part of the* ***core Who's On First dataset*** `https://github.com/whosonfirst-data` *but is instead a data from the* ***SFO Museum*** `https://github.com/sfomuseum-data` *modeled after, and compatible with, Who's On First.*

When I visit `http://localhost:8080/id/1477855657`, which is the ID for **Harvey Milk Terminal 1** `https://millsfield.sfomuseum.org/terminals/1477855657/` building at the San Francisco International Airport, the Browser application will render that ID like this:



	sfogis-ground
geom_alt	sfogis-subterranean

wof

belongsto	San Francisco International Airport
	85688637
	102191575
	85633793
	85922583
	102087579
	SFO Terminal Complex (2019)
	554784711
	102085387
breaches	–
controlled	wof:hierarchy
country	US
created	1522868103
geomhash	156a2cab536edb1ec070824f0923b462
hierarchy	**building_id** SFO Terminal Complex (2019)
	campus_id San Francisco International Airport
	continent_id 102191575
	country_id 85633793
	county_id 102087579
	locality_id 85922583
	postalcode_id 554784711
	region_id 85688637
	wing_id Terminal 1 (2019)
	building_id SFO Terminal Complex (2019)
	campus_id San Francisco International Airport
	continent_id 102191575
	country_id 85633793
	county_id 102085387
	region_id 85688637
	wing_id Terminal 1 (2019)
id	1477855657
label	Terminal 1 (2019)
lastmodified	2021-02-09T22:32:27.000Z
name	Terminal 1
parent_id	SFO Terminal Complex (2019)
placetype	wing
placetype_alt	terminal
repo	sfomuseum-data-architecture
superseded_by	Terminal 1 (2020-~-05)
supersedes	Terminal 1 (2017-~)
tags	terminal

status

is current	0	false	true
is ceased	1	true	true
is deprecated	-1	false	false
is superseded	1	true	true
is superseding	1	true	true

- edtf
- geom
- iso
- lbl
- mz
- name
- sfo
- sfomuseum
- src
- wof

If you look closely at that screenshot you'll see that most of the other places the terminal building is associated with don't have names. The exceptions are the terminal complex and the airport itself. That's because both of those records are part of the **sfomuseum-data-architecture** `https://github.com/sfomuseum-data/sfomuseum-data-architecture` repository which we've referenced in the `-reader-uri` flag at startup.

It's possible to specify multiple "reader" sources so if I restart the server telling it to read data from both the **sfomuseum-data-architecture** `https://github.com/sfomuseum-data/sfomuseum-data-architecture` and **sfomuseum-data-whosonfirst** `https://github.com/sfomuseum-data/sfomuseum-data-whosonfirst` repositories, like this:

```
$> bin/whosonfirst-browser \
        -reader-uri repo:///usr/local/data/sfomuseum-data-architecture \
        -reader-uri repo:///usr/local/data/sfomuseum-data-whosonfirst \
        -enable-all -nextzen-api-key {NEXTZEN_APIKEY}
```

I'll see the same thing but all those pointers to other places referenced by the Harvey Milk Terminal will have names now:

The Who's On First Browser can be configured to use a number of **different "reader" implementations** `https://github.com/whosonfirst/?q=go-reader` for consuming Who's On First data from a variety of sources. I'll write more about those readers, in detail, in future blog posts.

Under the hood the Who's On First Browser uses a library called **go-http-server** `https://github.com/aaronland/go-http-server` to abstract away the details of how it is configured to serve requests. Different server implementations are defined using custom URIs. For example to start the server as a standalone web server you would pass the following command-line flag:

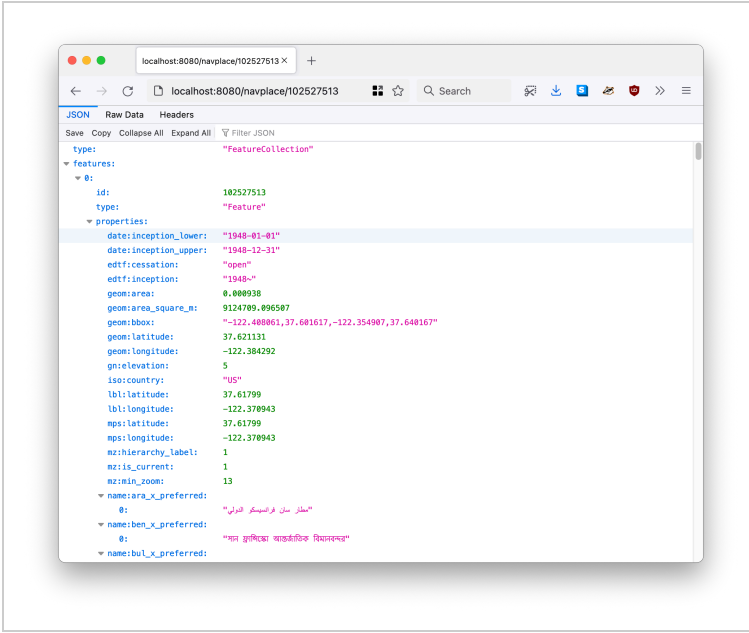
```
-server-uri http://localhost:8080
```

And to start the server as a Lambda function you would specify:

```
-server-uri lambda://
```

In point of fact, to start the server as a Lambda function you would actually set the `BROWSER_SERVER_URI=lambda://` environment variable which is then used to assign the `-server-uri` flag behind the scenes. But the idea is still the same: It should be easy to switch between deployment environments simply by changing the `-server-uri` flag.

And that's pretty much the story of the Who's On First Browser for the last few years. Every once in a while a new "representation" endpoint is added to the server, like the ability to produce **IIIF navPlace** `https://github.com/whosonfirst/go-whosonfirst-browser#iiif-navplace` records so you can reference places in Who's On First IDs from all your **IIIF manifests** `https://preview.iiif.io/api/navplace_extension/api/extension/navplace/`, but otherwise it hasn't been necessary to change anything.


localhost:8080/navplace/102527513
localhost:8080/navplace/102527513
Search
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON
type: "FeatureCollection"
features:
0:
id: 102527513
type: "Feature"
properties:
date:inception_lower: "1948-01-01"
date:inception_upper: "1948-12-31"
edtf:cessation: "open"
edtf:inception: "1948~"
geom:area: 0.000938
geom:area_square_m: 9124709.096507
geom:bbox: "-122.408061,37.601617,-122.354907,37.640167"
geom:latitude: 37.621131
geom:longitude: -122.384292
gn:elevation: 5
iso:country: "US"
lbl:latitude: 37.61799
lbl:longitude: -122.370943
mps:latitude: 37.61799
mps:longitude: -122.370943
mz:hierarchy_label: 1
mz:is_current: 1
mz:min_zoom: 13
name:ara_x_preferred:
0: "مطار سان فرانسيسكو الدولي"
name:ben_x_preferred:
0: "সান ফ্রান্সিস্কো আন্তর্জাতিক বিমানবন্দর"
name:bul_x_preferred:

Tailscale

Uniform patch: Pan American World Airways, Security police. Gift of the Pan Am Association. **Collection of SFO Museum**. 2000.058.0754

At the beginning of November, 2022 the **Tailscale** `https://tailscale.com/` group published a blog post titled **Virtual private services with tsnet** `https://tailscale.com/blog/tsnet-virtual-private-services/` . In that post they wrote:

> Tailscale lets you connect to your computers from anywhere in the world. We call this setup a virtual private network. Any device on the tailnet (our term for a Tailscale network) can connect directly to any other device on the tailnet. This isn't limited to your computers, phones, and servers, though. You can use Tailscale as a library in Go programs to allow them to connect to your tailnet as though it were a separate computer. You can also use Tailscale to run multiple services with different confidentiality levels on the same machine. Today I'm going to explain more about how you can use tsnet to make your internal services easier to run, access, and secure by transforming them into virtual private services on your tailnet.

Basically what all of that meant was that it would be possible to write an implementation of the `go-http-server` interfaces to allow web services to be started as standalone web servers, Lambda functions or Tailscale virtual private services. Instead of starting an instance of the Who's On First Browser like this:

```
-server-uri http://localhost:8080/
```

It could be started like this:

```
-server-uri tsnet://whosonfirst:80?auth-key={TAILSCALE_AUTH_KEY}
```

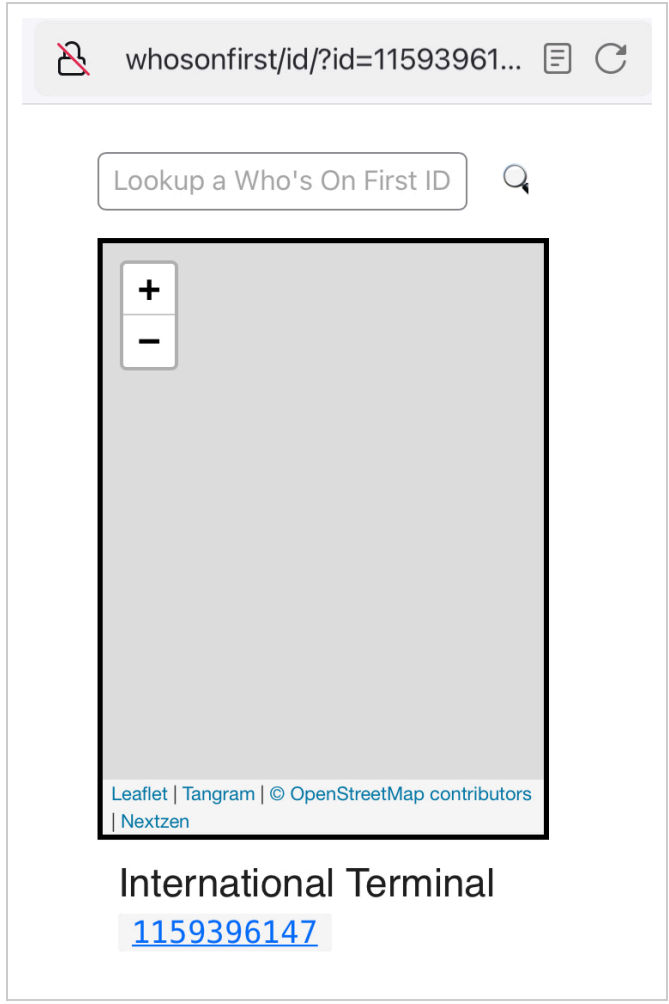
And that instance of the Who's On First Browser would be available in any web browser at `http://whosonfirst` but only to people logged in to the Tailscale VPN associated with `{TAILSCALE_AUTH_KEY}`. Importantly the computer running the machine running the Who's On First Browser doesn't need to be

publicly available, or reachable, on the internet at large which is, sadly, a practical consideration that you need to think about in 2022.

With all of that in mind, the **go-http-server-tsnet** `https://github.com/aaronland/go-http-server-tsnet` server implentation was written and has since been added to the **go-whosonfirst-browser** `https://github.com/whosonfirst/go-whosonfirst-browser` codebase. But why is this important?

It's important because not everyone using Who's On First, or a complimentary project like the **SFO Museum's Who's On First compatible dataset** `https://millsfield.sfomuseum.org/blog/tags/whosonfirst`, may not want to be as open or transparent as the Who's On First project in their own work. It's as simple as that. There are lots of different reasons why the access control and security that tools like Tailscale virtual private services provide may be desirable so we're happy to start making it part of the Who's On First tooling.

And then, while running the Who's On First browser on one computer and loading `http://whosonfirst` from a separate mobile device, connected to Tailscale, I noticed this:



What happened to the **Nextzen** `https://nextzen.org` map tiles? Were tiles not being displayed because the mobile device in question had **Lockdown Mode** `https://support.apple.com/en-us/HT212650` enabled and wouldn't enable WebGL (required to render Nextzen tiles using **Tangram.js** `https://github.com/tangrams/tangram`)? Were

tiles not being displayed because of **networking or DNS issues** `https://tailscale.com/learn/why-split-dns/` in Tailscale? Or... was this, instead, an opportunity to solve a completely different problem like adding support for **Protomaps** `https://protomaps.com` map tiles to the Who's On First Browser?

Yes!

Protomaps

Folding fan: Philippine Airlines 1960s. Gift of Thomas G. Dragges. **Collection of SFO Museum**. 2002.035.158

Protomaps is a suite of tool for producing digital maps consisting of

two parts:

1. ***A single static data file containing vector tile data, derived from OpenStreetMap*** `https://openstreetmap.org` ***, for a geographic region (or even the whole world) that can be served from a variety of cloud-based storage providers or just a plain old web server.***

2. ***Software which reads which that vector data by translating*** `{z}/{x}/{y}` ***map tile requests in to HTTP Range requests*** `https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Range` ***(so you don't have to download the whole data file) and styling the responses (using the HTML Canvas*** `https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API` ***APIs rather than WebGL).***

This is interesting for a few reasons:

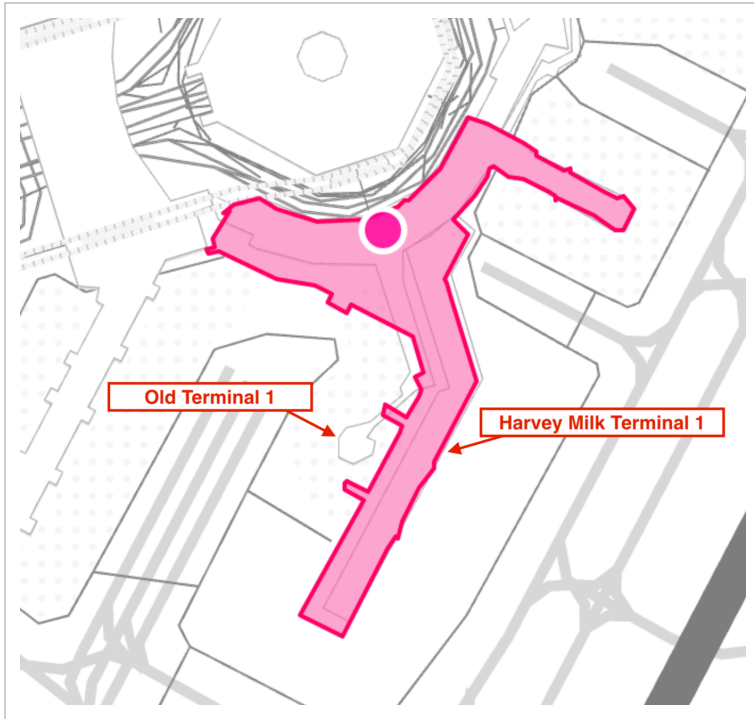
1. ***You might not want to use a third-party service, not even the good ones like Nextzen*** `https://nextzen.org` ***, to host your maps.***

2. ***You don't want to depend on a third-party service for maps but you also don't want to manage the burden of a global map-tile infrastructure.***

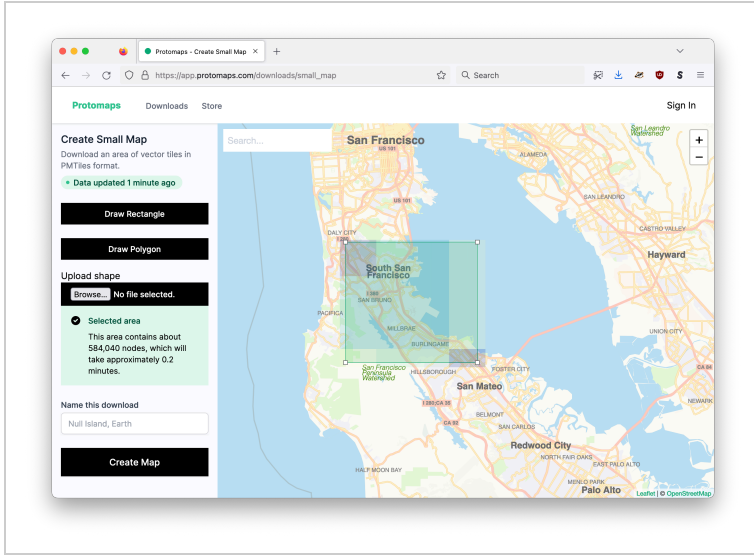
3. ***You don't want WebGL to be a requirement for rendering display maps.***

4. ***You want map data that is updated at the speed of the OpenStreetMap project.***

The Nextzen base tiles haven't been updated since 2019 which is a problem for places like SFO and SFO Museum since entire buildings that were there in 2019 have been demolished and replaced; for example the old Terminal 1 with its distinctive round boarding areas which is now the Harvey Milk Terminal 1 concourse. This is not just a Nextzen issue, either; this same problem exists today, in 2022, **in both Apple Maps and Google Maps** `https://millsfield.sfomuseum.org/blog/2022/08/08/maps/` .



So, where do Protomaps data files come from? You can either create your own, **pay for global data files** `https://app.protomaps.com/store` or use the free **Protomaps Small Map tool** `https://app.protomaps.com/store` to generate data files for small geographic areas. Here's what that looks like for the area around SFO:



This will produce a file called `sfo.pmtiles`. To use that data file with the Who's On First Browser you would specify the following command-line options:

```
$> bin/whosonfirst-browser/main.go \
        -enable-all \
        -reader-uri repo:///usr/local/data/sfomuseum-data-architecture \
        -reader-uri repo:///usr/local/data/sfomuseum-data-whosonfirst \
        -map-provider protomaps \
        -protomaps-bucket-uri file:///usr/local/data/ \
        -protomaps-tiles-database sfo
```

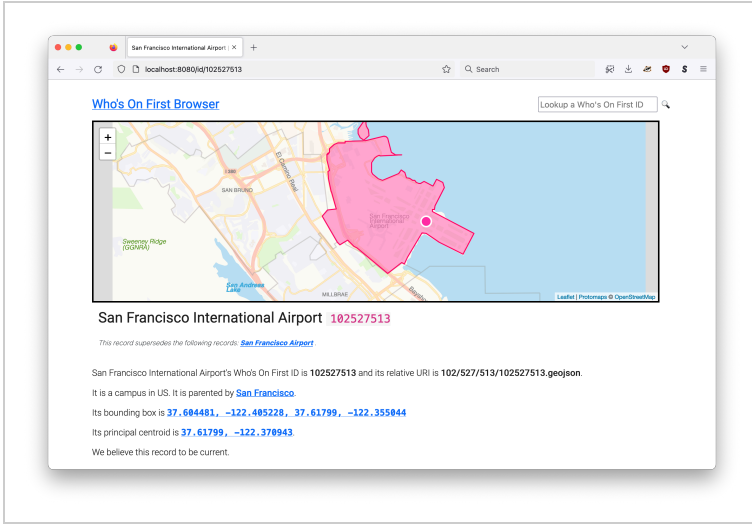
Do you see the various "protomaps" flags? This is what they mean:

1. ***`-map-provider protomaps` overrides the default map provider (Nextzen).***

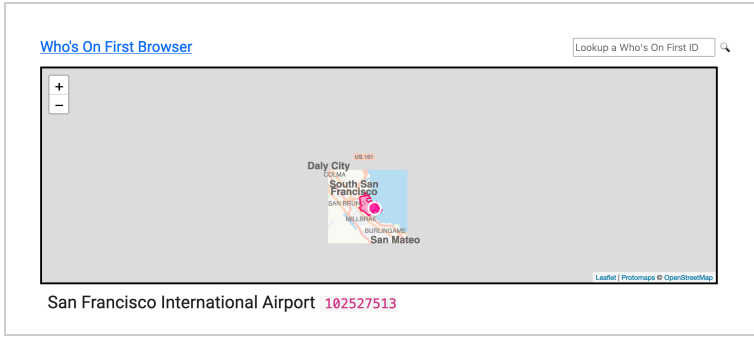
2. ***`-protomaps-bucket-uri file:///usr/local/data/` is fully-qualified URI where Protomaps data files are located.***

3. ***`-protomaps-tiles-database` is the name of the Protomaps data files to use for display maps.***

When configured this way the Who's On First Browser is serving tile requests using the data file specified by `protomaps-bucket-uri` and `protomaps-tiles-database`. For example:



Remember: There are only tiles for the immediate area around SFO because that's all the data the `sfo.pmtiles` file contains.



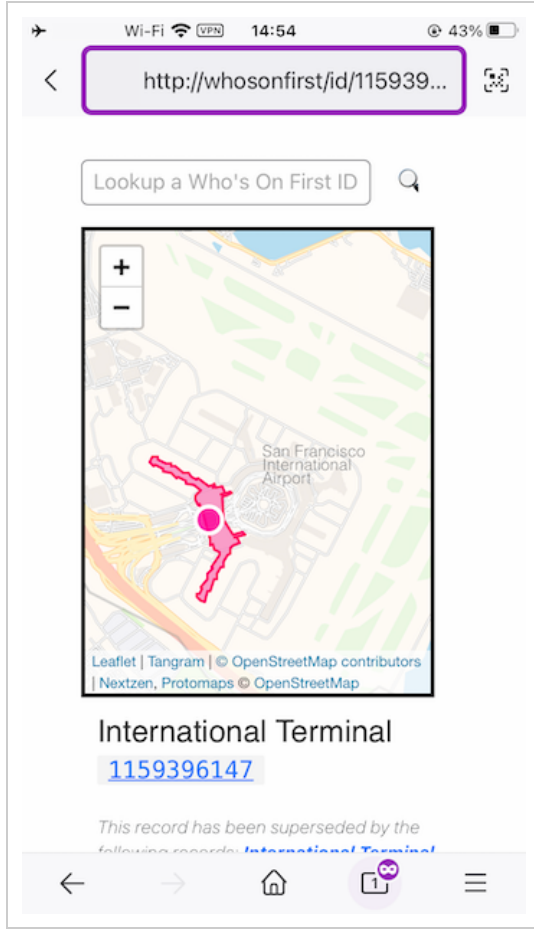
In these examples tiles are being served from data stored on the local disk but it is also possible to serve data stored in any remote location that can be read using the **GoCloud Blob** `https://gocloud.dev/howto/blob/` package, for example an **AWS S3 bucket** `https://gocloud.dev/howto/blob/#s3` .

Finally, to make it all work on a Tailscale network just add a `-server-uri tsnet://...` command-line flag:

```
$> bin/whosonfirst-browser/main.go \
        -enable-all \
        -reader-uri repo:///usr/local/data/sfomuseum-data-architecture \
        -reader-uri repo:///usr/local/data/sfomuseum-data-whosonfirst \
        -map-provider protomaps \
        -protomaps-bucket-uri file:///usr/local/data/ \
        -protomaps-tiles-database sfo \
        -server-uri tsnet://whosonfirst:80?auth-key={TAILSCALE_AUTH_KEY}

2022/11/11 22:25:47 Listening on http://whosonfirst:80
```

And here are the map tiles being displayed as expected in a mobile client connected to a Tailscale VPN visiting `http://whosonfirst/id/1159396147`:



Protomaps has **its own styling language** `https://protomaps.com/docs/frontends/leaflet#custom-vector-data` for defining how maps should be displayed. As of this writing custom styles are not supported by the Who's On First Browser but only because I haven't decided on the best way to import them. If you have some ideas or thoughts about how that should work **we'd love your**

input `https://github.com/whosonfirst/go-whosonfirst-browser` .

Postcard: Los Angeles International Airport 1960s. Gift of Thomas G. Dragges. **Collection of SFO Museum**. 2015.166.2054

As mentioned at the top of the post both of these features, running the Who's On First Browser as a Tailscale virtual private service and using Protomaps for display maps, are disabled by default but they are ready to be enabled when the circumstances present themselves.

2022-11-14

things I have written about elsewhere #2022129

A global point-in-polygon service using a static...

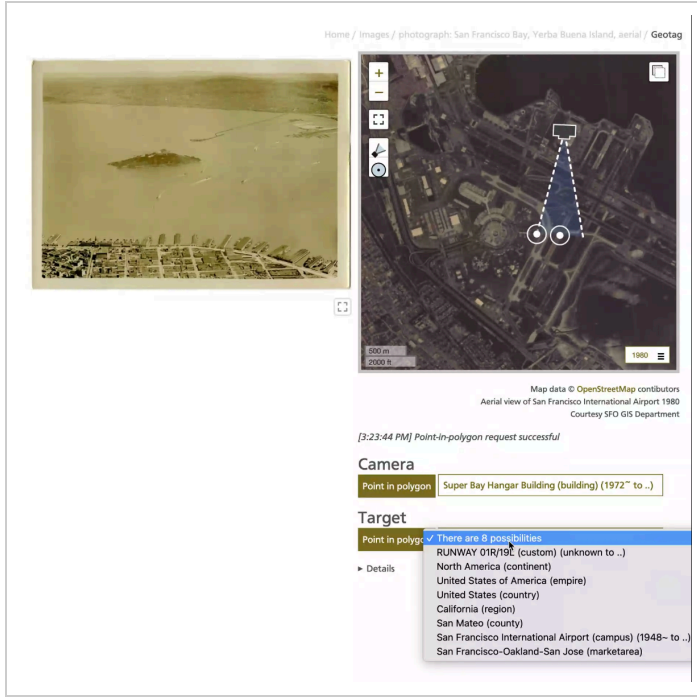
A global point-in-polygon service using a static 8GB data file

This was **originally published on the SFO Museum Mills Field weblog** `https://millsfield.sfomuseum.org/blog/2022/12/19/pmtiles-pip/` , in December 2022.

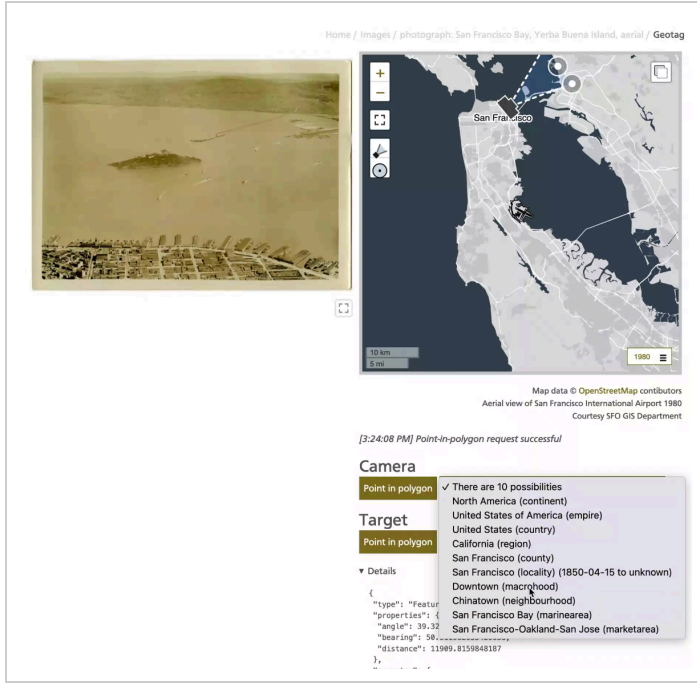
Here's the punchline: A global point-in-polygon service that returns Who's On First records and costs a few dollars a month to run. To explain how that's possible let's start with a video.

This video `images/sfomuseum-pmtiles-point-in-polygon.mp4` shows some in-progress work we're doing to develop interfaces for geotagging objects in our collection. This example shows **an aerial photo of Yerba Buena Island and the San Francisco Bay** `https://collection.sfomuseum.org/objects/1511942515/` taken by Stanley Henry Page in 1922. Note the absence of the Bay Bridge which wouldn't be completed for another 14 years! All of this work builds on **the many geotagging-related blog posts** `https://millsfield.sfomuseum.org/blog/tags/geotagging` written in 2020 and, in particular, our use of the **New York Public Library's Leaflet.GeotagPhoto** `https://millsfield.sfomuseum.org/blog/2020/04/23/geotagging/` interface control to define not just where an image was taken but also the field of view emcompassed by that image.

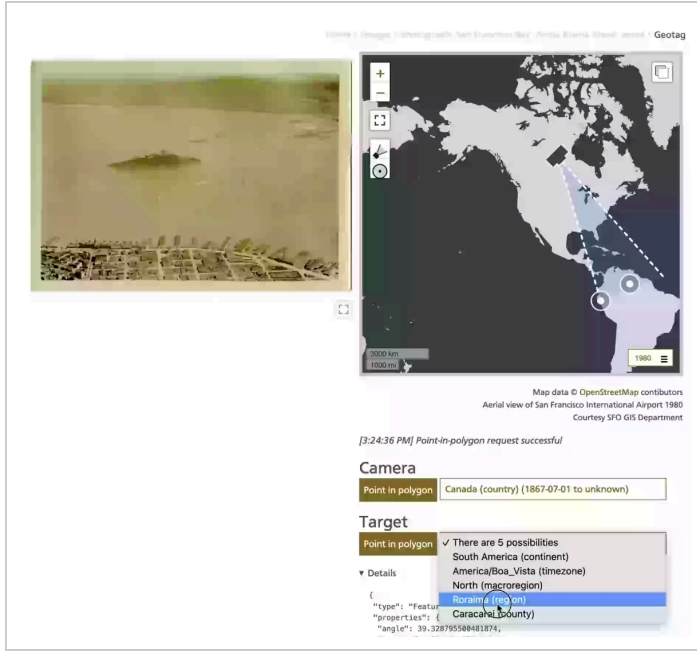
Recently, we've been working on the tools to "reverse-geocode" those two points. Reverse-geocoding assigns stable, permanent identifiers for the named places represented by a pair of latitude and longitude coordinates. We've written about this before in the **Reverse-Geocoding in Time at SFO Museum** `https://millsfield.sfomuseum.org/blog/2021/03/26/spatial/` and **Geotagging at SFO Museum: Protomaps, search and reverse-geocoding** `https://millsfield.sfomuseum.org/blog/2021/05/03/geotagging/` blog posts. In both cases we were working with location data and databases scoped to SFO and the San Francisco Bay Area but SFO Museum's collection spans the entire planet so the challenge has been to think about inexpensive and low-overhead ways to make reverse-geocoding possible for any place on Earth. The results of those efforts are what you're seeing in the video above.



The video starts by showing the image of Yerba Buena next to a map. The default starting point for objects that haven't been geotagged is the SFO campus so the first thing you see is a demonstration of what happens as the camera and field of view controls are moved around. Each time a control stops moving its latitude and longitude is sent to a "reverse-geocoding" server and the list of places associated with that point are used to populate a drop-down menu. This allows us to say that an image (not this image) was taken "from" the **Super Bay Hangar** `https://millsfield.sfomuseum.org/id/1477855969/` building and that it "depicts" **Runway 01R/19L** `https://millsfield.sfomuseum.org/id/1730008753/` .



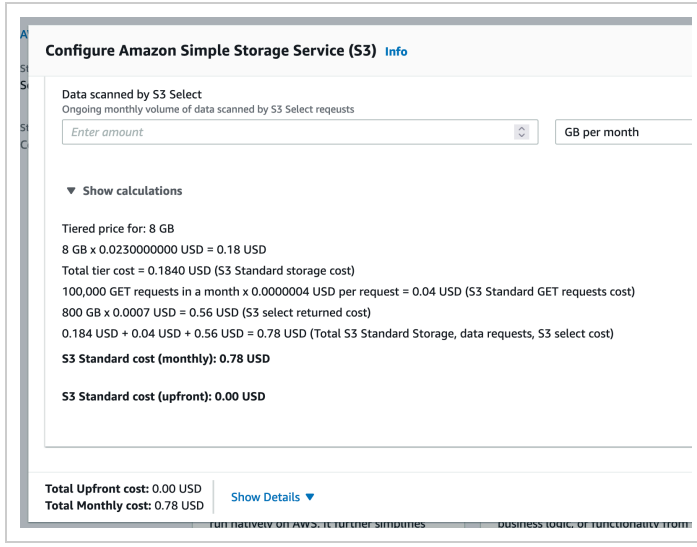
Next the camera icon is moved up to the City of San Francisco, and positioned where we think the photograph (of Yerba Buena island) was taken, and the field of view control is extended to stretch across the San Francisco Bay. Downtown San Francisco and the City of Berkeley are selected as the locations where the image was taken and what it depicts, respectively. Is this correct? The photo wasn't taken "from" downtown San Francisco so much as above it and the focus of the photo is really Yerba Buena more than Berkeley. But, as you can see from the video, there are a number of places to choose from and we can move the field of view control to better suit our understanding of an image. Failing a consensus about the exact locations we could always say the photo depicts San Francisco and Alameda counties or, if nothing else, at least the state of California.



Afterwards, the map is zoomed out to display the Americas. The camera is moved up to Canada and the field of view control to **the Roraima region in Brazil** `https://spelunker.whosonfirst.org/id/85682011/` and then across the ocean to Africa and finally to Central Europe where there are both contemporary and historical records for the countries that made up **the former Yugoslavia** `https://whosonfirst.org/blog/2017/06/29/tackling-space-and-time-in-whosonfirst/` . These unrealistic camera views are just to demonstrate that it's possible to perform point-in-polygon queries *anywhere in the world*. As a place is selected from the results of a point-in-polygon query its unique ID, and all the unique IDs of its ancestors, is appended to the coordinate data for the camera and the field of view controls. These are data we then append to the object records in our collection which enables us to search for objects associated with not just a specific location but all the other places that parent it.

These point-in-polygon requests are being made to a server that has been deployed using the **API Gateway in front of a Lambda function** `https://docs.aws.amazon.com/whitepapers/latest/serverless-multi-tier-architectures-api-gateway-lambda/web-application.html`

pattern. That's a pretty common pattern these days and not terribly exciting (or at least "novel", I think it's still pretty exciting). The exciting part is that the server is executing the point-in-polygon queries against a static 8GB datafile stored in an S3 bucket rather than a traditional spatial database. That static datafile contains the entirety of the **Who's On First administrative corpus** `https://github.com/whosonfirst-data?q=whosonfirst-data-admin-&type=all&language=&sort=` in addition to the **catalog of architectural features at the San Francisco International Airport** `https://github.com/sfomuseum-data/sfomuseum-data-architecture` and was created using openly-licensed data and open-source software.



A screenshot of the AWS Cost Estimate Calculator showing estimated costs for hosting an 8GB file using their S3 storage service.

This datafile can be stored and served from an AWS S3 bucket for as little as $0.18/month to as much as $9/month if, unlikely as it may be, you were to serve 8000GB worth of data out of that datafile in a month. So, really, a global point-in-polygon service that returns Who's On First records for about $1/month. All of this builds on work we wrote about in the **Reverse-Geocoding in Time at SFO Museum** `https://millsfield.sfomuseum.org/blog/2021/03/26/spatial/` blog post. In fact the server piece described in that blog post is the same one being used in this example. The difference is that instead of storing data and performing

spatial queries in a SQLite database we are storing data in a remote Protomaps datafile and reading small slices of that file, rather than the whole thing, in order to extract just the information necessary to perform a spatial query in memory.

So, what is "a Protomaps database"? **Protomaps** `https://protomaps.com` describes itself as "a serverless system for planet-scale maps". It is a suite of tools and a database format for efficiently storing, retrieving and rendering map tile data at a fraction of the size and cost of other map-related infrastructures. The Protomaps database format employs clever means to organize and efficiently retrieve map tile data but that data is, ultimately, just a collection of compressed GeoJSON features. This means we don't have to store data designed specifically for rendering cartographies but can store other geographic data like Who's On First place records. For any given tile there are only a finite number of places and their geometries will have been trimmed to the boundary of the tile so, assuming data has been stored at zoom 12 tiles or higher, the amount of data to transfer out of a Protomaps database will be small (or at least small enough). All of this makes it possible, given a latitude and longitude coordinate, to:

1. *Derive the map tile associate with that coordinate (at a specified zoom level)*

2. *Retrieve just that tile from a Protomaps database without, and this is important, needing to download or read the entire database*

3. *Load the GeoJSON data contained in that tile in to an in-memory spatial data structure*

4. *Query that data for features whose geometry contains the coordinate*

This approach is not as fast as other approaches. Notably it is nowhere near as fast as an actual spatial database. However, when you factor in the cost and the overhead of running and maintaining a dedicated spatial database, combined with a variety of strategies for caching results, it is very often fast enough. This is not the only way we could have solved the problem of a global-scale reverse-geocoding service but we are happy to be able to have done so in a way that puts this kind of functionality within the reach of people and organizations who might not otherwise be in a position to deploy, or maintain, the traditional infrastructure used to solve these

kinds of problems.

Postcard: Eastern Air Lines, Douglas. Paper, ink. Gift of Thomas G. Dragges, SFO Museum Collection. ***2015.166.0636***

How does it work?

This section documents the different data and software packages which make all of this possible. This is the Very Technical part of the blog post so if that's not relevant to your interests you can stop here knowing that something which, at the scale of every place on Earth, used to be hard and expensive is now cheap, easy and possible.

The rest of this blog post will document each of the steps used to create a Protomaps database full of Who's On First records and the query it using both a command line tool and an HTTP web service.

Who's On First data

Poster: various airlines. Paper, ink. Gift of Robert McCrory, SFO Museum Collection. ***2010.157.003***

Who's On First data is the raw geographic data that we're going to encode, for example one or more data repositories from the **whosonfirst-data** `https://github.com/whosonfirst-data` or the **sfomuseum-data** `https://github.com/sfomuseum-data` GitHub organizations.

1. ***https://github.com/whosonfirst-data*** `https://github.com/whosonfirst-data`

2. ***https://github.com/sfomuseum-data*** `https://github.com/sfomuseum-data`

whosonfirst/go-whosonfirst-tippecanoe

Photograph: San Francisco International Airport (SFO), Central Terminal construction. Photograph. Transfer from San Francisco International Airport, SFO Museum Collection. ***2005.137.050.002***

go-whosonfirst-tippecanoe `https://github.com/whosonfirst/go-whosonfirst-tippecanoe` is a software package that will iterate through one or more Who's On First data repositories, filter out records whose geometry is not a polygon, trim the record properties to a minimum, common set of properties (optionally appending additional properties you choose to specify) and emitting everything as line-separated GeoJSON records.

For example, here's how you might create line-seperated GeoJSON records for features in the **sfomuseum-data-whosonfirst** `https://github.com/sfomuseum-data/sfomuseum-data-whosonfirst` and **sfomuseum-data-architecture** `https://github.com/sfomuseum-data/sfomuseum-data-architecture` respositories:

```
$> bin/features \
        -as-spr -require-polygons \
        -spr-append-property wof:hierarchy \
        -writer-uri 'constant://?val=featurecollection://?writer=stdout://' \
        /usr/local/data/sfomuseum-data-whosonfirst/ /usr/local/data/sfomuseum-data-architecture \
        > sfomuseum.jsonl
```

Do you see the `-spr-append-property wof:hierarchy` flag that's being included in the command above? This tells the tool to append the `wof:hierarchy` property of each feature to the final output. We'll revisit that later in this blog post.

1. ***https://github.com/whosonfirst/go-whosonfirst-tippecanoe*** `https://github.com/whosonfirst/go-whosonfirst-tippecanoe`

felt/tippecanoe

Swizzle stick: American Airlines. Plastic. Gift of Thomas G. Dragges, SFO Museum Collection. ***2001.116.189***

tippecanoe `https://github.com/felt/tippecanoe` is a software package that consumes GeoJSON records and encodes their data as map tiles in **an "MBTiles" database** `https://github.com/mapbox/mbtiles-spec` . For example, here's how you might do that for the `sfomuseum.jsonl` file created in the last example, encoding all the data in tiles at zoom level 13.

```
$> tippecanoe -P -z 13 -pf -pk -o sfomuseum.mbtiles sfomuseum.jsonl
```

`tippecanoe` is also able to read data from STDIN to rather than creating an intermediary `sfomuseum.jsonl` file you can also just pipe the output of the `features` tool straight to `tippecanoe`. For a good discussion of how tippecanoe works take a look at Erica Fischer's **How We Make Your Data Look Great At Every Scale with Tippecanoe** `https://felt.com/blog/tippecanoe-`

`display-scale-precision` blog post.

1. ***https://github.com/felt/tippecanoe*** `https://github.com/felt/tippecanoe`

protomaps/go-pmtiles

Playing cards: Western Airlines. Cardboard, plastic, ink. Gift of Thomas G. Dragges, SFO Museum Collection. ***2002.095.044***

go-pmtiles `https://github.com/protomaps/go-pmtiles` is a software package that will convert a "MBTiles" database of map tiles in to a Protomaps database (of map tiles). Here's how you would convert the 'sfomuseum.mbiles' database created in the last example in to a Protomaps database.

```
$> pmtiles convert sfomuseum.mbtiles sfomuseum.pmtiles
```

1. ***https://github.com/protomaps/go-pmtiles*** `https://github.com/protomaps/go-pmtiles`

whosonfirst/go-whosonfirst-spatial-pmtiles

Brochure: Northeast Airlines. Paper, metal, ink. Gift of the William Hough Collection, SFO Museum Collection. ***2008.055.992***

go-whosonfirst-spatial-pmtiles `https://github.com/whosonfirst/go-whosonfirst-spatial-pmtiles` is a software package for performing point-in-polygon queries against a Protomaps database (of map tiles). In addition to library code for performing those queries it also provides tools for building a Protomaps database of Who's On First data in a **Docker** `https://docker.com` container and a command-line tool for doing point-in-polygon queries against a local, or remote, Protomaps database.

docker

The **Dockerfile** `https://github.com/whosonfirst/go-whosonfirst-spatial-pmtiles/blob/main/docker/Dockerfile` included with the package wraps all the steps described above and can be run locally or in a hosted container service. For example, here's how you might build the same Protomaps database, that we've created in the examples above, storing the output locally, in a folder named `/usr/local/data`.

```
$> docker build -t whosonfirst-spatial-pmtiles .

$> docker run -v /usr/local/data:/usr/local/data whosonfirst-spatial-pmtiles  \
        /usr/local/bin/build.sh \
        -n sfomuseum \
        -z 13 \
        -s 'sfomuseum-data://?prefix=sfomuseum-data-whosonfirst sfomuseum-data://?prefix=sfomuseum-d
```

Note that this particular example might take a long time depending on things like bandwidth speeds and processing power; anything map-related over zoom 12 starts to take a long time. It is also written specifically to read source data from repositories stored in a GitHub organization, rather than data stored locally. There are also **example configuration files** `https://github.com/whosonfirst/go-whosonfirst-spatial-pmtiles/tree/main/docker/ecs` for running this container using Amazon's Elastic Container Service.

1. ***https://github.com/whosonfirst/go-whosonfirst-spatial-pmtiles/tree/main/docker*** `https://github.com/whosonfirst/go-whosonfirst-spatial-pmtiles/tree/main/docker`

query

The `query` tool is a command-line tool for performing point-in-polygon queries against a Protomaps database. Here's how you would query the `sfomuseum` database, created in examples, above for a point at SFO. In this example we're also asking that the results be sorted first by placetype, then by name and finally by the inception date. The output of that query is run through the **jq** `https://stedolan.github.io/jq/` tool to produce a simple list containing each place's name, inception date and placetype.

```
$> ./bin/query \
        -spatial-database-uri 'pmtiles://?tiles=file://usr/local/data&database=sfomuseum' \
        -latitude 37.615761 \
        -longitude -122.389154 \
        -sort-uri placetype:// \
        -sort-uri name:// \
        -sort-uri inception:// \

        | jq '.places[] | "\(."wof:name") \(."edtf:inception") (\(."wof:placetype"))"'
```

```
"North America  (continent)"
"United States  (country)"
"California  (region)"
"San Mateo  (county)"
"San Francisco International Airport 1948~ (campus)"
"SFO Terminal Complex 2000~ (building)"
"SFO Terminal Complex 2006~ (building)"
"SFO Terminal Complex 2011~ (building)"
"SFO Terminal Complex 2014~ (building)"
"SFO Terminal Complex 2017~ (building)"
"SFO Terminal Complex 2019-07-23 (building)"
"SFO Terminal Complex 2020-~05 (building)"
"SFO Terminal Complex 2021-05-25 (building)"
"SFO Terminal Complex 2021-11-09 (building)"
"International Terminal 2000~ (wing)"
"International Terminal 2006~ (wing)"
"International Terminal 2011~ (wing)"
"International Terminal 2020-~05 (wing)"
"International Terminal 2021-05-25 (wing)"
"International Terminal 2021-11-09 (wing)"
"International Terminal 2000~ (wing)"
"International Terminal 2006~ (wing)"
"International Terminal 2011~ (wing)"
"International Terminal 2014~ (wing)"
"International Terminal 2017~ (wing)"
"International Terminal 2020-~05 (wing)"
"International Terminal 2021-05-25 (wing)"
"International Terminal 2021-11-09 (wing)"
"International Terminal 2000~ (wing)"
"International Terminal 2006~ (wing)"
"International Terminal 2011~ (wing)"
"International Terminal 2014~ (wing)"
"International Terminal 2017~ (wing)"
"International Terminal 2019-07-23 (wing)"
"International Terminal 2020-~05 (wing)"
"International Terminal 2021-05-25 (wing)"
"International Terminal 2021-11-09 (wing)"
"International Terminal Main Hall 2017~ (concourse)"
"International Terminal Main Hall 2019-07-23 (concourse)"
"International Terminal Main Hall 2020-~05 (concourse)"
"International Terminal Main Hall 2021-05-25 (concourse)"
"International Terminal Main Hall 2021-11-09 (concourse)"
```

As you can see any given point on the SFO campus could have been one of any number of different concourses or buildings over time. We've written about this in detail in the **Who's On First at SFO Museum** `https://millsfield.sfomuseum.org/blog/2018/08/28/whosonfirst/` and the **Reverse-Geocoding in Time at SFO Museum** `https://millsfield.sfomuseum.org/blog/2021/03/26/spatial/` blog posts. Although it's not shown in this example it's also possible to filter point-in-polygon queries using fuzzy or ambiguous dates encoded using the Library of Congress's **Extended DateTimeFormat** `https://millsfield.sfomuseum.org/blog/2021/01/14/edtf/` . You can also see how we are able to pair SFO Museum's **bespoke data** `https://github.com/sfomuseum-data` alongside the core **Who's On First dataset** `https://github.com/whosonfirst-data` .

1. ***https://github.com/whosonfirst/go-whosonfirst-spatial-pmtiles#query*** `https://github.com/whosonfirst/go-whosonfirst-`

spatial-pmtiles#query

whosonfirst/go-whosonfirst-spatial-www-pmtiles

Postcard: Pan American World Airways. Paper, ink. Gift of Thomas G. Dragges, SFO Museum Collection. ***2015.166.1213***

go-whosonfirst-spatial-www-pmtiles `https://github.com/whosonfirst/go-whosonfirst-spatial-www-pmtiles` is a software package for running a web service that allows point-in-polygon queries to be performed against a Protomaps database using HTTP requests. This server can be deployed as a standalone HTTP service or using **the "API Gateway in front of a Lambda function" pattern** `https://github.com/whosonfirst/go-whosonfirst-spatial-www-pmtiles#lambda` . Here's how you might start that service, running locally, using the Protomaps database created in the examples above.

```
$> bin/server \
        -server-uri http://localhost:8080 \
        -spatial-database-uri 'pmtiles://?tiles=file:///usr/local/data&database=sfomuseum&enable-cac
        -properties-reader-uri '{spatial-database-uri}' \
```

The service is queried by sending it a JSON-encoded **point-in-polygon request** `https://github.com/whosonfirst/go-whosonfirst-spatial-pip/blob/main/pip.go#L13-L28` to its `/api/point-in-polygon` endpoint. For example:

```
{
   "latitude" : 37.615761,
   "longitude" : -122.389154,
   "properties" : [
      "wof:hierarchy"
```

```
   ],
   "sort" : [
      "placetype://",
      "name://",
      "inception://"
   ]
}
```

Before going any further there are a few things to note:

1. ***The JSON-encoded point-in-polygon request contains a `properties: [ "wof:hierarchy" ]` property. This tells the server to append a `wof:hierarchy` property to each result. Where does that data come from? It comes from a "properties reader" instance defined by the `-properties-reader-uri` flag.***

2. ***There can be multiple different properties readers but in this example the value of `{spatial-database-uri}` signals that the underlying spatial database is also the place where properties are stored; the spatial database is also the properties reader. Remember the `-spr-append-property wof:hierarchy` flag defined in an earlier example? That's what added these non-standard data to the Protomaps database which we're going to append to our point-in-polygon results.***

3. ***The `-spatial-database-uri` flag itself contains a `?enable-cache=true` query parameter. This signals that individual features should be cached as they are retrieved from the underlying map tiles in the Protomaps database. This local cache allows additional properties to be appended to the final results of a spatial query.***

Eventually, all of these steps will be consolidated but they are necessary today. Putting it all together we end up with something like this:

```
$> curl -X POST http://localhost:8080/api/point-in-polygon \
       -d '{"latitude": 37.615761, "longitude": -122.389154, "properties": [ "wof:hierarchy" ], "so
```

```
{
  "places": [
    {
      "edtf:cessation": "",
      "edtf:inception": "",
      "mz:is_ceased": -1,
      "mz:is_current": 1,
      "mz:is_deprecated": -1,
      "mz:is_superseded": 0,
      "mz:is_superseding": 0,
      "mz:latitude": 0,
```

```
    "mz:longitude": 0,
    "mz:max_latitude": -122.433347,
    "mz:max_longitude": -122.342023,
    "mz:min_latitude": 37.578044,
    "mz:min_longitude": 0,
    "wof:belongsto": [],
    "wof:country": "",
    "wof:id": "102191575",
    "wof:lastmodified": 1652218109,
    "wof:name": "North America",
    "wof:hierarchy":[
      {
          "continent_id":102191575
      }
    ],
    "wof:parent_id": "-1",
    "wof:path": "102/191/575/102191575.geojson",
    "wof:placetype": "continent",
    "wof:repo": "sfomuseum-data-whosonfirst",
    "wof:superseded_by": [],
    "wof:supersedes": []
  },
  ...and so on
```

1. ***https://github.com/whosonfirst/go-whosonfirst-spatial-www-pmtiles#server*** `https://github.com/whosonfirst/go-whosonfirst-spatial-www-pmtiles#server`

Reading remote datafiles

Serving tray: United Airlines; McDonnell Douglas DC-10. Plastic, paint. Gift of the McAfee family in memory of Robert E. McAfee, SFO Museum Collection. ***2015.126.121***

So far all the examples have demonstrated reading data from a local Protomaps database. It is also possible to read data from a Protomaps database hosted on a remote storage service like Amazon's S3. Under the hood the value of the `?tiles=` query parameter in the `-spatial-database-uri` flag is just a **gocloud.dev/blob** `https://gocloud.dev/howto/blob/` URI so data can be read from any source that implements that package's **Blob** `https://pkg.go.dev/gocloud.dev/blob` interfaces. The default supported URIs are `file://` for reading data from the local filesystem and `s3://` and `s3blob://` for reading data from an S3 bucket. The difference between `s3://` and `s3blob://` is that the latter allows AWS credentials to be **defined as a query parameter** `https://github.com/aaronland/gocloud-blob-s3#credentials` . For example:

```
s3blob://{S3_BUCKET}?region={AWS_REGION}&prefix={PREFIX}&credentials={CREDENTIALS}
```

A concrete example, reading data from an S3 bucket called `example` in a folder called `tiles/` in the AWS `us-east-1` region, specifying an AWS credentials profile called `example` would be:

```
s3blob://example?region=us-east-1&prefix=tiles/&credentials=example
```

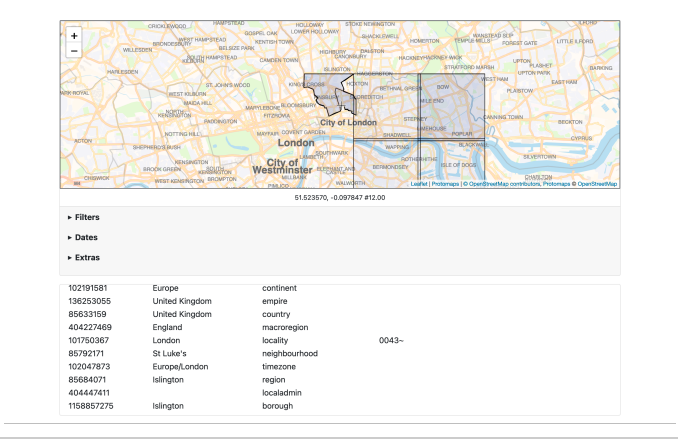
Unfortunately, because the URI contains its own query parameters, it needs to be URL-encoded before it is included in the final `-spatial-database-uri` URI. Putting it altogether we end up with this:

```
-spatial-database-uri 'pmtiles://?tiles=s3blob%3A%2F%2Fexample%3Fregion%3Dus-east-1%26prefix%3Dtiles
```

Which is definitely not pretty but it does the job, provides a simple declarative syntax to toggle between local and remote data sources and is something that can be improved on in future releases.

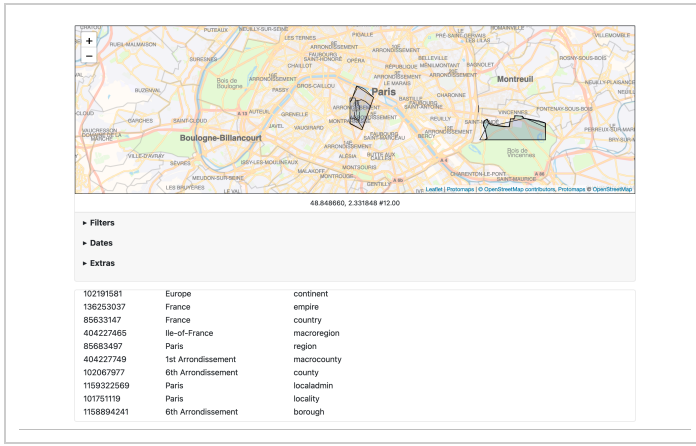
Postcard: San Francisco International Airport (SFO), The International Room. Paper, ink. Gift of Thomas G. Dragges, SFO Museum Collection. ***2015.166.2287***

A few final words about cached data



It is possible to tell the `server` tool in the **go-whosonfirst-spatial-www-pmtiles** `https://github.com/whosonfirst/go-whosonfirst-spatial-www-pmtiles#server` package to return results as a set of GeoJSON records. This functionality is not really suited for use with a Protomaps database. That's because the geoemtries for any given place are trimmed to the boundaries of the map tile that contains it. Large areas, like regions or countries or continents, will simply be

squares unless you happen to be doing a point-in-polygon query near a coastline and even smaller places will be trimmed in they happen to span multiple tiles. Compounding the issues, as of this writing, is that the local caching mechanism is storing feature records using only their Who's On First identifiers independent of the tile currently being queried. As you can see in these screenshots that yields some puzzling results. An improved caching strategy is on the to-do list but, ultimately, if you are using a Protomaps database for doing point-in-polygon requests *and* you need complete geometries for the places returned in a query request you will need to fetch them in separate requests from `https://data.whosonfirst.org/geojson/{ID}` or `https://static.sfomuseum.org/geojson/{ID}` or **an equivalent data source** `https://github.com/whosonfirst/go-whosonfirst-browser2` .



Update

Since publishing this blog post there have been some questions about whether or not SFO Museum is operating a public point-in-polygon service. At this time we are not. However, in the spirit of generousity we have produced and published a public Protomaps database file containing all of the

administrative records in Who's On First `https://github.com/whosonfirst-data/?q=whosonfirst-data-admin&type=all&language=&sort=` at zoom level 12. The file itself is 3.8GB and unfortunately we need to reserve the right to remove it, at any time, if the bandwidth costs become an issue. Maybe it won't but, sadly, in 2022 it's always a possibility. For the time being here's how you can download and use the file locally:

First, start by downloading the file using a tool like `wget` or equivalent. In these example it is assumed the local file is saved to `/usr/local/data/whosonfirst.pmtiles`.

```
$> mkdir -p /usr/local/data
$> wget -O /usr/local/data/whosonfirst.pmtiles https://static.sfomuseum.org/pub/point-in-polygon/who
```

If you just want to query the database file from the command line download a copy of the **go-whosonfirst-spatial-pmtiles** `https://github.com/whosonfirst/go-whosonfirst-spatial-pmtiles` package. The only thing you'll need to install to compile the `query` tool is a copy of the **Go compiler** `https://go.dev/dl/` which will build the tool for any of the Windows, Apple or Linux operating systems.

```
$> cd go-whosonfirst-spatial-pmtiles
$> go build -mod vendor -o bin/query cmd/query/main.go
```

And then query the database just like you would in the examples above:

```
$> ./bin/query \
        -spatial-database-uri 'pmtiles://?tiles=file:///usr/local/data/&enable-cache=true&database=w
        -latitude 37.615761 \
        -longitude -122.389154 \
        -sort-uri placetype:// \
        | jq '.places[]["wof:name"]'

2022/12/20 11:28:35 time to index paths (0) 20.53µs
2022/12/20 11:28:35 GET tile at /whosonfirst/12/655/1585.mvt
2022/12/20 11:28:35 fetching whosonfirst 0-16384
2022/12/20 11:28:35 Prune tile cache older that 2022-12-20 11:23:35.357902 -0800 PST m=-299.97137776
2022/12/20 11:28:40 fetched whosonfirst 0-0
2022/12/20 11:28:40 fetching whosonfirst 3003051-11907
2022/12/20 11:28:40 fetched whosonfirst 3003051-11907

"North America"
"United States of America"
"United States"
"California"
"San Mateo"
"San Francisco International Airport"
"San Francisco-Oakland-San Jose"
"America/Los_Angeles"
```

If you want to query the database file from a web service download the **go-whosonfirst-spatial-www-pmtiles** `https://github.com/whosonfirst/go-`